



Pontificia Universidad
JAVERIANA
Cali

**Facultad de Ingeniería
y Ciencias**
Ingeniería Electrónica

TRABAJO DE GRADO

Optimización del Control y la Gestión de Datos en una
Celda de Manufactura Flexible a través de una Base de
Datos Orientada a Grafos

Isabella Ceballos Sánchez
Juan José Restrepo Rosero

Director
Dr. Juan David Contreras Pérez

Codirector
Dr. María Constanza Pabón Burbano

8 de diciembre de 2023

Santiago de Cali, 8 de diciembre de 2023

Señores
Pontificia Universidad Javeriana – Cali
Dr. Hernán Camilo Rocha Niño
Decano
Facultad de Ingeniería y Ciencias
Cali

Cordial Saludo.

Por medio de la presente nos permitimos presentarle el Trabajo de Grado titulado “Optimización del Control y la Gestión de Datos en una Celda de Manufactura Flexible a través de una Base de Datos Orientada a Grafos”.

Esperamos que este trabajo reúna todos los requisitos académicos, cumpla el propósito para el cual fue creado y sirva de apoyo para futuros proyectos relacionados con la materia.

Atentamente,

Isabella Ceballos S.

Isabella Ceballos Sánchez

Juan José Restrepo R.

Juan José Restrepo Rosero

Santiago de Cali, 8 de diciembre de 2023

Señores

Pontificia Universidad Javeriana – Cali

Dr. Hernán Camilo Rocha Niño

Decano

Facultad de Ingeniería y Ciencias

Cali

Cordial Saludo.

Certificamos que el presente Trabajo de Grado titulado “Optimización del Control y la Gestión de Datos en una Celda de Manufactura Flexible a través de una Base de Datos Orientada a Grafos”, realizado por Isabella Ceballos Sánchez y Juan José Restrepo Rosero, estudiantes de Ingeniería Electrónica, se encuentra terminado y puede ser presentado para su sustentación.

Atentamente,



Dr. Juan David Contreras Pérez
Director Trabajo de Grado



Dr. María Constanza Pabón Burbano
Co-Director Trabajo de Grado

Dedicatoria

El presente trabajo de grado se lo dedicamos a Dios, familiares y a amigos cercanos que nos han acompañado en esta experiencia de formación como Ingenieros Electrónicos.

Agradecimientos

Agradecemos especialmente a nuestros directores de grado, que han estado muy pendientes del progreso, nos han servido de soporte y guía para la realización de este proyecto. Agradecemos especialmente su constante apoyo durante los casi 7 meses de desarrollo e implementación del proyecto.

Además, agradecemos a todos los docentes de la carrera, que a partir de sus enseñanzas, nos permitieron lograr la formación como Ingenieros Electrónicos. Agradecemos a nuestras familias quienes nos han apoyado en este proceso con sus consejos, además, han permitido que podamos seguir un buen camino para su realización y han estado constantemente pendiente del desarrollo de nuestro trabajo de grado.

Resumen

Los actuales sistemas de gestión de bases de datos (DBMS) desempeñan el papel de organizar los datos de manera eficiente y eficaz. Sin embargo, entre los desarrollos aplicados, sigue existiendo un factor importante relacionado al esquema de representación y análisis de los datos, en donde es vital tener en cuenta la manera en cómo se almacenan los datos, debido a que esto influye en la eficiencia de su procesamiento y consulta.

Una celda de manufactura flexible, junto con los sistemas de ejecución de manufactura (MES), están orientados hacia la optimización de operaciones de un área de trabajo, con el objetivo de asegurar la ejecución efectiva y optimización del rendimiento de las operaciones de fabricación de la planta de producción, y mejorar la calidad del producto final. Estos sistemas, funcionan a partir del control y análisis de la información recopilada.

Por ende, la implementación de una base de datos en el MES se convierte en uno de los pilares claves del control de un proceso, es decir, el constante flujo de datos obtenidos debe ser regulado y analizado eficazmente para generar valor y mejorar la eficiencia, aprovechando al máximo la representación digital del proceso. Por lo tanto, es de vital importancia conocer las diferentes bases de datos que se han desarrollado actualmente, y analizar sus diferentes desempeños en cuanto a su flexibilidad, escalabilidad y adaptabilidad a los cambios de la celda.

Así pues, en este trabajo se aborda un estudio en torno al mejoramiento de los procesos de control y la gestión de datos de una celda de manufactura flexible, mediante el uso de base de datos orientada a grafos y herramientas de visualización para consultar, visualizar y entender los indicadores definidos para el proceso.

Palabras Claves: Base de datos de grafos, Celdas de manufactura, Neo4j, Cypher, Sistemas de ejecución de manufactura.

Abstract

Current database management systems (DBMS) play the role of organizing data efficiently and effectively. However, among the developments applied, there is still an important factor related to the data representation and analysis scheme, where it is vital to take into account the way in which data is stored, because this influences the efficiency of its processing and querying.

A flexible manufacturing cell, together with manufacturing execution systems (MES), are oriented towards the optimization of operations of a work area, with the objective of ensuring the effective execution and performance optimization of the manufacturing operations of the production plant, and improving the quality of the final product. These systems work from the control and analysis of the information collected.

Therefore, the implementation of a database in the MES becomes one of the key pillars of process control, i.e., the constant flow of data obtained must be effectively regulated and analyzed to generate value and improve efficiency, making the most of the digital representation of the process. Therefore, it is of vital importance to know the different databases that have been currently developed, and to analyze their different performances in terms of their flexibility, scalability and adaptability to cell changes.

Thus, this paper addresses a study on the improvement of control processes and data management of a flexible manufacturing cell, through the use of graph-oriented database and visualization tools to query, visualize and understand the indicators defined for the process.

Keywords: Graph database, Manufacturing cells, Neo4j, Cypher, Manufacturing execution systems.

Índice general

1. Introducción	1
2. Descripción del Problema	3
3. Justificación	5
4. Objetivos	7
4.1. Objetivo General	7
4.2. Objetivos Específicos	7
5. Marco Teórico	8
5.1. Términos clave	8
5.1.1. Celda de Manufactura	8
5.1.2. MES (Manufacturing Execution System)	8
5.1.3. Trazabilidad	9
5.1.4. Indicadores Industriales	9
5.2. Bases de Datos	11
5.2.1. Entorno de Operación de un Sistema de Gestión de Bases de Datos(SGBD) . .	11
5.2.2. Base de datos Relacionales (SQL)	12
5.2.3. Base de datos No Relacionales (NoSQL)	14
5.2.4. Bases de Datos Relacionales VS NoSQL	16
5.2.5. Base de datos orientada a grafos	17
5.3. Protocolos de Comunicación Industriales	24
5.3.1. OPC UA (Arquitectura Unificada de Comunicaciones de Plataforma Abierta)	24
5.3.2. Ethernet Industrial	25
5.3.3. Modbus TCP/IP	25
5.3.4. Comunicación Serial RS232 y señales digitales I/O	26
5.4. Estado del Arte	28
5.4.1. Wireless cyber-physical systems performance evaluation through a graph database approach	28
5.4.2. Lenguaje visual de consulta basado en transformación de grafos: aplicación en el dominio médico	28
5.4.3. Topology Modeling and Analysis of a Power Grid Network Using a Graph Database	28
5.4.4. Bases de datos orientadas a grafos aplicadas al estudio de informes radiológicos: utilizando entornos de computación en la nube para abordar estudios de gran dimensión	29

5.4.5. A knowledge graph-based data representation approach for IIoT-enabled cognitive manufacturing	29
6. Definición de Requisitos	30
6.1. Descripción del Caso de Estudio	30
6.1.1. Estado actual del sistema	30
6.1.2. Bloques de Conexión	34
6.1.3. Lógica Funcional	35
6.2. Especificación de Requisitos	39
7. Diseño del Sistema de Información	41
7.1. Base de Datos	41
7.1.1. Diseño conceptual	41
7.1.2. Depuración del diseño	44
7.1.3. Boceto Final:	45
7.2. Dashboard	49
7.3. Interfaz Gráfica (GUI)	53
8. Implementación del Sistema de Información	57
8.1. Base de Datos	58
8.2. Dashboard: Looker Studio	61
8.3. Interfaz Gráfica (GUI): Tkinter Python	67
8.4. Simulación Celda de Manufactura del CAP: RoboDK	68
8.4.1. Configuración de RoboDK	68
8.5. Software de Control	70
8.6. Validación de Conexión y Comunicación (Software de Control - RoboDK - Dashboard - Base de Datos y GUI)	73
8.6.1. Pruebas Botón Crear Orden	73
8.6.2. Pruebas Botón Modificar Orden	77
8.6.3. Pruebas Botón Eliminar Orden	81
8.6.4. Pruebas Botón Ver Órdenes	84
9. Operación del Sistema de Información	86
9.1. Pruebas Botón Activar Celda de Manufactura	86
9.2. Pruebas Botón Re-abastecer Almacén	91
10. Dificultades	94
11. Conclusiones y Trabajos Futuros	98
11.1. Conclusiones	98
11.2. Trabajos futuros	99
12. Anexos	100

Índice de figuras

5.1.	Jerarquía Funcional sistema MES.[1]	9
5.2.	ANSI-SPARC Arquitectura de tres niveles.[2]	12
5.3.	Bases de datos NoSQL[3].	15
5.4.	Modelo de datos y desafíos bases SQL y NoSQL. [4]	17
5.5.	Base de datos de grafos, ilustración de una red de informes médicos. [5]	18
5.6.	Base de datos de referencia que muestra la información de supervisión y citas para investigadores, estudiantes y publicaciones [6]	21
5.7.	Consulta en Cypher [6]	21
5.8.	Resultado de línea 2 (a), línea 3 (b)[6]	21
5.9.	Resultado de línea 2 - 4 [6]	22
5.10.	Resultado de línea 5[6]	23
5.11.	Resultado de línea 6 y 7 [6]	23
6.1.	FMC del CAP	30
6.2.	Almacén ASRS	31
6.3.	Estación de torneado	32
6.4.	Estación de Fresado	32
6.5.	Estación de Montaje	33
6.6.	Diagrama Global de la Celda de Manufactura del CAP	34
6.7.	Diagrama de Comunicación Ethernet Celda de Manufactura del CAP	34
6.8.	Diagrama de comunicación OPC UA Celda de Manufactura del CAP	35
6.9.	Lógica Funcional Celda de manufactura CAP Parte 1	37
6.10.	Lógica Funcional Celda de manufactura CAP Parte 2	38
7.1.	Boceto No.1 Base de Datos	42
7.2.	Boceto No.2 Base de Datos	44
7.3.	Boceto No.3 Base de Datos	46
7.4.	Boceto No.4 Base de Datos	48
7.5.	Boceto Reporte de Órdenes	50
7.6.	Boceto Reporte Estación Torno	51
7.7.	Boceto Reporte Estación Fresado	51
7.8.	Boceto Reporte Estación ASRS	52
7.9.	Boceto Reporte Estación Inspección	52
7.10.	Boceto de Crear Orden	53
7.11.	Boceto de Eliminar Orden	53
7.12.	Boceto de Modificar Orden	54
7.13.	Boceto de Ver Órdenes	54

7.14. Boceto de Re-establecer Almacén	55
7.15. Boceto de Activar Celda de Manufactura	55
7.16. Boceto de Botón de Emergencia/ Pantalla Inicio	56
8.1. Boceto Comunicación Componentes del Sistema de Información	57
8.2. Implementación del Boceto No.4 en Neo4j	58
8.3. Atributos Nodo Máquinas	59
8.4. Atributos Nodo Piezas	60
8.5. Atributos Nodo Estaciones	60
8.6. Atributos Nodo Material	61
8.7. Reporte de Órdenes	62
8.8. Reporte de Estación Torno	63
8.9. Reporte de Estación Fresado	64
8.10. Reporte de Estación ASRS	65
8.11. Reporte de Estación Inspección	65
8.12. Reporte de Banda Transportadora	66
8.13. Interfaz Gráfica GUI desarrollada en Tkinter Python	67
8.14. Celda de Manufactura en RoboDK	69
8.15. Estaciones en RoboDK	70
8.16. Lógica funcional Software de Control Parte 1	71
8.17. Lógica funcional Software de Control Parte 2	72
8.18. Secuencia de pasos para Crear una Orden	74
8.19. Descripción de ID Orden	74
8.20. Actualización Base de Datos	75
8.21. Atributos Nodo Tipo Orden	75
8.22. Actualización Nodo Material	76
8.23. Actualización Dashboard	77
8.24. Secuencia de pasos para Modificar una Orden	78
8.25. Actualización Material Empack	79
8.26. Actualización Orden EP2_2023_23_10_C3_H9_T33	79
8.27. Actualización relación PIECE	80
8.28. Actualización Dashboard	80
8.29. Secuencia de pasos para Eliminar una Orden	82
8.30. Actualización Base de Datos al Eliminar Orden	83
8.31. Actualización Material Empack	83
8.32. Actualización Dashboard al Eliminar Orden	84
8.33. Secuencia de pasos para ver las Órdenes	85
9.1. Secuencia de pasos para Ejecutar Celda	86
9.2. Actualización Base de Datos	87
9.3. Actualización de piezas producidas	87
9.4. Actualización Dashboard Órdenes	88

9.5. Actualización Dashboard Estación Torno	89
9.6. Actualización Dashboard Estación Fresado	89
9.7. Actualización Dashboard Estación Inspección	90
9.8. Actualización Dashboard Estación ASRS	90
9.9. Actualización Dashboard Bandas Transportadoras	90
9.10. Secuencia de pasos para Re-abastecer Almacén	91
9.11. Actualización Material Aluminio	92
9.12. Actualización Material Empack	92
9.13. Re-abastecer Almacén RoboDK	93
10.1. Daño en la batería del Torno CNC	94
10.2. Daño en el controlador del eje Z del ASRS	95
10.3. Daño en la correa de la banda transportadora	95
10.4. Daño en la correa de la banda transportadora	96
10.5. Daño en la correa del ASRS	96

CAPÍTULO 1

Introducción

Las crecientes demandas de productos personalizados por parte de los clientes, plantean un gran desafío para las empresas manufactureras en términos de flexibilidad y capacidad de respuesta [7]. Hoy en día, se han propuesto diversos enfoques de programación dinámica eficaces para que los sistemas de fabricación respondan rápidamente a los cambios en las demandas de los usuarios, sin embargo, la implementación de un método de programación automática con alta precisión de control y bajo retardo de control sigue siendo un desafío el cual sigue siendo un tema en desarrollo donde se integran nuevas tecnologías y métodos que requieren de una gran cantidad de trabajo y tiempo para ajustar un esquema de programación que pueda responder a las necesidades de los procesos de manufactura actuales.

Como respuesta a esta problemática, en [8] se propone una arquitectura de control adaptativo de dos niveles habilitada por un método de programación automática e integrada en una celda de fabricación usando gemelos digitales (DTMC por sus siglas en inglés), teniendo como objetivo asignar automáticamente un esquema de programación de tareas de entrada con un lote de trabajos en un grupo de programas de control a través de una red de modelos de comportamiento y un conjunto de modelos de eventos integrados en la DTMC, para adaptarse con mayor velocidad al ajuste dinámico del esquema de programación causado por los cambios en la demanda de los clientes o en las condiciones de producción.

Actualmente, en los procesos de celda de manufactura también se evidencia el uso de aplicaciones de procesamiento y almacenamiento de datos mediante un sistema de gestión de bases de datos (DBMS). En [9] se describen implementaciones comunes de DBMS relacionales y No-SQL. Se comparan sus capacidades mediante un ejemplo práctico, teniendo como resultado que las bases de datos relaciones tienen un rendimiento deficiente en cuanto a su escalabilidad y proceso de algunos tipos de datos. Del mismo modo, en [10] propone un método de almacenamiento de Big Data (BD) para redes de distribución inteligente de un sistema de energía basado en la base de datos de grafos Neo4j. Los resultados de la simulación en los sistemas de prueba estándar IEEE muestran que el método de almacenamiento de BD propuesto puede mejorar de manera efectiva el almacenamiento de datos y aumentar la eficiencia del análisis de la red de distribución inteligente.

En particular, las consultas sobre modelos de grafos han cobrado relevancia en los últimos años debido a su aplicación en áreas como el análisis de datos biológicos, las redes sociales y la web semántica. Por otra parte, no es nada raro encontrar hoy en día, que el ofrecer sistemas de información que cuenten con herramientas que faciliten el acceso y consultas de datos, es un gran reto

que se sigue enfrentando desde diferentes perspectivas. En [11], la autora propone un lenguaje visual de consulta sobre un modelo de grafos, enfocado en el usuario final, que ofrece mayor expresividad que las herramientas de exploración de grafos sin trasladar a una notación visual los elementos de un lenguaje basado en texto. Como resultado de este trabajo, se obtuvo que el lenguaje facilita la formulación de consultas ad hoc (no conocidas con anticipación) que pueden ser complejas, mostrando en las pruebas que es una opción muy viable, teniendo en cuenta el tiempo de ejecución, en comparación con otros sistemas de bases de datos.

CAPÍTULO 2

Descripción del Problema

Un sistema de ejecución de manufactura (MES), es un sistema que permite el control, monitoreo y sincronización de la ejecución de procesos industriales, mediante la gestión de datos recopilados[12]. En la implementación de un sistema MES, el análisis de datos juega un papel relevante en la mejora de la operación del ambiente de fabricación, en la calidad de los productos y en los reportes de lo sucedido en cada proceso. Para lograr esto, se requiere proporcionar flexibilidad para realizar cambios sobre la marcha de pedidos, especialmente en donde la producción de bienes es masiva [13][14].

Por un lado, en la actualidad es común encontrar que, los DBMS relacionales se usen para manejar diferentes volúmenes de datos. Este tipo de bases, aunque logran un alto desempeño en procesamiento, almacenamiento y tiempo de respuesta de consulta, los DBMS no relaciones, específicamente la de grafos, son más eficientes al generar consultas que implican un recorrido topológico [9]. Por ende, este trabajo se abordará el estudio de otro tipo de base de datos, en este caso, NoSQL, que se ajuste a los diferentes cambios en el proceso, y del mismo modo, provea robustez y escalabilidad al sistema, buscando facilitar las consultas requeridas en la base de datos.

Por otro lado, en comparación con las bases de datos relacionales, las DBMS No relacionales, también conocidas como No-SQL, han demostrado obtener mejores resultados en términos de escalabilidad y procesamiento de grandes volúmenes de datos, facilitando la adaptación de la base de datos a los cambios topológicos y lógicos del sistema de manufactura. Por esta razón, se decidió investigar más a fondo estos tipos de bases No-SQL, y en estudios previos como [10], [13], [15] y [16], donde se evidencia que la base de datos No-SQL que más se ajusta a las necesidades de la implementación de una celda de manufactura flexible es la orientada a grafos. Esto se debe a que la base de datos orientada a grafos se centra en la teoría de grafos, lo que permite representar la celda de manufactura flexible como una serie de nodos y etiquetas. La flexibilidad que brinda la representación por grafos, permite modelar la topología de los sistemas de manufactura y el sistema de información de forma integrada donde, por ejemplo, las estaciones de fabricación se pueden representar como nodos al igual que los pedidos u órdenes, y así relacionarlos entre sí por medio de aristas que contienen la información o etiquetas. Esta representación de los datos y sus relaciones permite un entendimiento, análisis y desarrollo de procesos mucho más simple en comparación con otros sistemas de bases de datos, que específicamente en el campo de los MES, funcionan como la representación de la topología del proceso y la relación que existe entre un elemento y otro.

En [17] los autores hacen un seguimiento de los sistemas de gestión de bases de datos de grafos (GDBMS) y exploran su aplicación en el dominio biomédico. Estos concluyen que, aunque los siste-

mas de administración de bases de datos relacionales (RDBMS) y otros motores NoSQL son útiles en cuestión de escalabilidad, realización de consultas y posibilidad de adición de nuevos datos, su rendimiento puede verse afectado en conjuntos de datos densamente conectados con relaciones de muchos a muchos. En estos casos, los GDBMS pueden ser beneficiosos al proporcionar un modelado más natural de las múltiples relaciones y ofrecer medios más intuitivos para la realización de consultas.

Teniendo en cuenta lo anterior, surge la siguiente pregunta: ¿Cómo se pueden mejorar los procesos de control y de gestión de los datos de una celda de manufactura flexible, mediante el uso de una base de datos que otorgue flexibilidad al proceso? En otras palabras, como respuesta a esta problemática, se propone un sistema de información para una celda de manufactura flexible mediante el uso de bases de datos orientadas a grafos, de tal manera que nos permita mejorar los procesos de control y gestión de los datos, obteniendo así un mejor entendimiento de sus iteraciones y recopilación de datos, y, por otro lado, la facilidad de permitir realizar cambios al proceso de celda de manufactura flexible brindando adaptabilidad al proceso.

CAPÍTULO 3

Justificación

En los últimos años, la transformación y digitalización del sector manufacturero, ha implicado una transición del paradigma industrial actual, en donde el mayor de los problemas se centra en encontrar maneras de lograr la interoperabilidad entre el mundo físico y el mundo digital de los sistemas de fabricación, para poder cubrir las demandas y mantener un ritmo de rápido crecimiento económico. Esto provoca que se propongan diversas soluciones en cuanto a programación dinámica, monitoreo, control y gestión de los datos, flexibilidad y capacidad de respuesta del proceso. [18].

Una celda de manufactura flexible se encuentra en constantes cambios en el entorno, es decir, aparecen nuevas estaciones, nuevas máquinas, nuevos procesos e incluso se re-configura para ejecutar otro proceso industrial que va acorde a la situación actual que exige el mercado a la empresa [19].

A partir de las mejoras de rendimiento y eficiencia de procesos de manufactura, se propone diseñar y aplicar un sistema de información orientado en grafos que permita controlar la ejecución de una celda de manufactura flexible y así conseguir que el sistema de información sea flexible a la celda de manufactura, además de obtener una gestión de la información que se recolecta para llevar un registro del rendimiento de la máquina y otros posibles indicadores industriales del proceso.

En el ámbito económico, un sistema MES al actuar como un sistema de control y de monitoreo, que gestiona los procesos de producción, permite que se asegure la ejecución efectiva y máxima optimización ya sea en recursos o rendimiento de las operaciones, del mismo modo, se obtiene una mejora en la calidad del producto final. A su vez, al obtener un producto de calidad, esto generará que los sistemas de planificación de recursos empresariales (ERP, por sus siglas en inglés) influyan sobre las decisiones de negocio.

En términos de resultados, a partir del planteamiento de la celda de manufactura flexible basada en una base de grafos, se genera oportunidades en la analítica y gestión de los datos debido a su similitud o su facilidad de representar la topología del proceso. Del mismo modo, en un trabajo futuro se podría realizar un análisis de los datos en cuanto a la planificación de la cadena de suministros de un sistema, es decir, se podría identificar una serie de patrones, y con base en esto, representar mejor el entorno y, por ende, tomar decisiones óptimas e incluso estimar o anticipar una falla y/o mantenimiento de la celda.

Principalmente, se espera que, al implementar un sistema de información orientado a grafos en una celda manufactura flexible, se obtenga una valorización de la información recolectada, a partir

de indicadores industriales, un control y gestión de los datos y otorgar flexibilidad en cuanto a los cambios repentinos del proceso.

CAPÍTULO 4

Objetivos

4.1. Objetivo General

Diseñar un sistema de información basado en una base de datos orientada a grafos, para el control y gestión de los datos en la celda de manufactura flexible del Centro de Automatización de Procesos (CAP).

4.2. Objetivos Específicos

- Definir los requerimientos a partir de trabajos relacionados y de la funcionalidad actual de la celda de manufactura flexible del CAP.
- Diseñar un sistema de información que permita el control y gestión de los datos de la celda de manufactura flexible del CAP a partir de una base de datos orientada a grafos.
- Implementar el sistema de información y un aplicativo de visualización que permita el control y la gestión de los datos.
- Evaluar el sistema de información propuesto para el control de la celda de manufactura del CAP en un escenario real o simulado de fabricación sobre pedido.

Marco Teórico

5.1. Términos clave

5.1.1. Celda de Manufactura

Las celdas de manufactura son un tipo de tecnología que está orientado hacia la optimización de operaciones de un área de trabajo, con el objetivo de eliminar de manera consistente y progresiva aquellas actividades que no poseen ningún valor agregado en el proceso de manufactura. Esta puede estar constituida por tantas máquinas y equipos que sean necesarios para la operación y desarrollo del proceso industrial. Es común encontrar que en ellas se encuentran un almacén que se utiliza para alojar tanto piezas por procesar, como piezas procesadas, un robot cartesiano que permite alcanzar las distintas localidades del almacén, un sistema de transporte que permite trasladar las piezas hacia el robot y de regreso hacia el almacén, y un manipulador robótico articulado que realiza un procesamiento [20].

Para una buena implementación de una celda de manufactura, es importante centrarse en cuestiones técnicas como su diseño, estructura, composición y funcionamiento. Pero, también debe centrarse en cuestiones humanas, debido a que las personas operarán, administrarán, apoyarán y sobre todo mantendrán las celdas en su correcto funcionamiento [21].

5.1.2. MES (Manufacturing Execution System)

El Sistema de Ejecución de Manufactura (MES), es un software que actúa como un sistema de control y de monitoreo de la información para gestionar procesos de producción en entornos industriales, documentando la gestión de la planta. Estos sistemas tienen como objetivo asegurar la ejecución efectiva y optimización del rendimiento de las operaciones de fabricación de la planta de producción, y mejorar la calidad del producto final [22].

El sistema MES requiere de sistemas tanto de control, como de supervisión y adquisición de datos (sistema SCADA, por sus siglas en inglés), con el fin de gestionar y controlar los equipos implementados. Del mismo modo, es fundamental integrar soluciones ERP (planificación de recursos empresariales, por sus siglas inglés) para transferir información sobre el rendimiento de la producción, el consumo de materiales y otros.

Un sistema MES se alimenta en tiempo real de los datos provenientes de otros sistemas, convirtiéndolos en información necesaria para la toma de decisiones [1]. A continuación, en la imagen 5.1, se muestra la jerarquía funcional de un sistema MES, en ella se encuentra que el MES cuenta con un nivel de planificación, donde recibe las órdenes de trabajo y logística del proceso, para poder así supervisar y controlar la información del proceso de producción.



Figura 5.1: Jerarquía Funcional sistema MES.[1]

5.1.3. Trazabilidad

La trazabilidad se entiende como la manera de encontrar y seguir el rastro de un producto o proceso en cada una de sus etapas de ejecución e identificar las condiciones que lo rodean a lo largo de la cadena de logística y detectar el origen de una incidencia con facilidad [23][24].

5.1.4. Indicadores Industriales

Un indicador industrial sirve para la medición cualitativa o cuantitativa de los resultados del proceso, equipo o de la propia empresa. Con el propósito de mejorar la eficacia de un MES, los fabricantes necesitan comprender las realidades sobre la Celda de Manufactura y así poder actuar ante inconvenientes si se desea contar con un MES fuerte que ayude a los trabajadores a responder mejor ante estas realidades. Ante esta situación, se puede aprovechar y explotar un recurso valioso en las plantas de fabricación: los datos de la máquina. Estos datos, al ser tenidos en cuenta, ofrecen un flujo de información en tiempo real que se puede utilizar para permitir una mejor toma de decisiones sobre la producción, una gestión más eficaz de los procesos y la utilización de la capacidad de la maquinaria.

A partir de [25], se lograron identificar algunos indicadores industriales que sirven de referencia para la gestión de datos y el monitoreo de la celda de manufactura del CAP:

1. **Eficacia General del Equipo (OEE):** Es una medida aproximada de cuán eficiente está siendo con su proceso. Para calcular el OEE, se deben medir tres puntos de datos: el tiempo de actividad del equipo (disponibilidad), la eficiencia del equipo mientras está en funcionamiento (eficiencia) y la cantidad de desechos producidos (calidad).

Si se desea alcanzar un puntaje OEE del 100 por ciento, se deben fabricar solo piezas buenas tan rápido como la máquina pueda funcionar sin tiempo de inactividad (Downtime en inglés), pero, en la mayoría de las industrias un OEE perfecto no es posible, debido a la fuerte caída en el ROI a medida que se acerca al 100% y por ello, una puntuación del 40% al 60% se considera típica y el 85% se considera de clase mundial para la mayoría de las industrias.

2. **Tiempos de Operación:** Esta métrica permite comprender la cantidad de horas utilizadas para realizar actividades en un período determinado (un día, una semana, un mes, etc.) en producción. El resultado de esta métrica permite saber si las horas dedicadas son más de las esperadas, lo que puede indicar ineficiencias y posibles áreas de mejora.
3. **Tiempos de Inactividad (Downtime):** Al rastrear estos incidentes y reportar esta información al gerente de la planta en lugar de solo al operador de la máquina, se pueden identificar las máquinas que impiden que la línea de producción funcione de manera eficiente y así, tomar en cuenta dicha información para abordar los problemas relacionados a las máquinas o procesos y aportar a la toma de decisiones de posibles mejoras.
4. **Número de productos producidos:** Este indicador mide la cantidad de productos fabricados en un período específico. Se puede utilizar de forma más restringida, por ejemplo, para conocer la productividad de un sector, empleado o turno de trabajo en un intervalo de tiempo.

5.2. Bases de Datos

Una base de datos es un conjunto de datos relacionados entre sí, diseñados para satisfacer las necesidades de información de una organización. Es la representación de un conjunto estructurado de datos, que contienen físicamente el diseño lógico de un conjunto de entidades del sistema de información que se está modelando en una organización [2].

En este documento, se estudiarán a cerca de los tipos de bases de datos relacionales (SQL) y las no relacionales (NoSQL).

5.2.1. Entorno de Operación de un Sistema de Gestión de Bases de Datos(SGBD)

Es un sistema que permite crear, editar y controlar el acceso a una base de datos. Estos se componen de: [2]

- Hardware (Contenedor de datos).
- Software (Programa de aplicación).
- Datos (Átomos de la base de datos).
- Procedimientos (Reglas que gobiernan las bases de datos).
- Usuarios (Personas que interactúan con la base de datos).

Por otro lado, las organizaciones American National Standard Institute (ANSI) y Standard Planning and Requirements Committee (SPARC), plantean una arquitectura de 3 niveles para un SGDB, permitiendo la descripción global de la base de datos. En la figura 5.2, se ilustra los 3 niveles propuestos por (ANSI-SPARC), detallando la información y el propósito del nivel [2]:

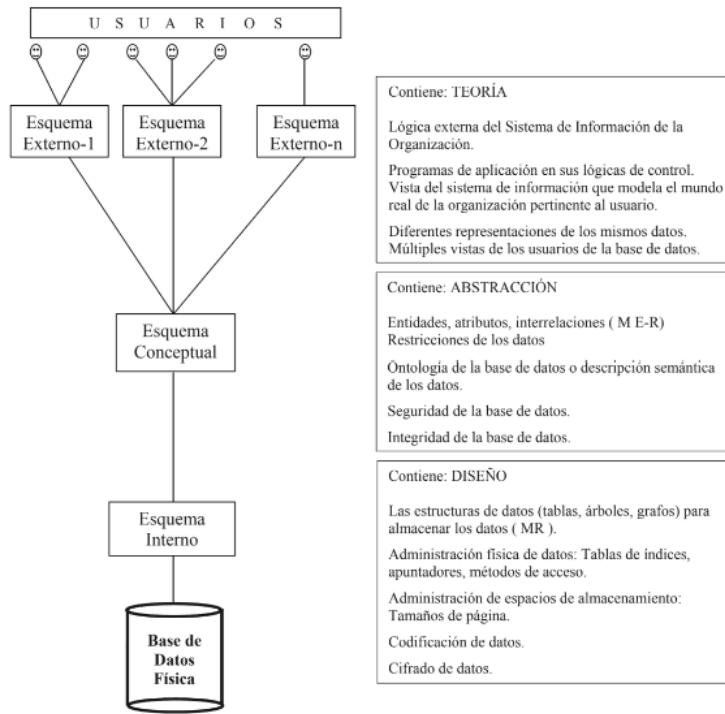


Figura 5.2: ANSI-SPARC Arquitectura de tres niveles.^[2].

5.2.2. Base de datos Relacionales (SQL)

Una base de datos relacional, es un tipo de base de datos que almacena y proporciona acceso a puntos de datos relacionados entre sí. Estos datos se representan como un conjunto de tablas con columnas y filas, en donde se guarda información sobre los objetos que se van a representar en la base de datos. Las columnas contienen los atributos de los datos y cada registro o fila, suele tener un valor o ID único relacionado a cada atributo, lo que simplifica la creación de relaciones entre los puntos de datos. ^[26] ^[27].

Fueron definidas por Edgar Frank Codd a finales de los 60, con los siguientes objetivos: ^[28]

- **Independencia física:** La forma de almacenar los datos no debe de influir en su manipulación lógica.
- **Independencia lógica:** Las herramientas que hacen uso de las bases de datos, no deben de modificarse si la base de datos se modifica.
- **Flexibilidad:** Busca ofrecer distintas vistas en función de los usuarios y aplicaciones.
- **Uniformidad:** Las bases de datos cuentan con una única forma conceptual (Tablas).

5.2.2.1. 12 Reglas de Codd

Buscan regir las bases de datos para ser consideradas relacionales. Estas reglas consisten en: [28]

1. **Información:** Toda la información de la base de datos debe estar representada explícitamente en el esquema lógico. Es decir, todos los datos están en las tablas.
2. **Acceso garantizado:** Todo dato es accesible sabiendo el valor de su clave y el nombre de la columna o atributo que contiene el dato.
3. **Tratamiento sistemático de los valores nulos:** El DBMS debe permitir el tratamiento adecuado de estos valores.
4. **Catálogo en línea basado en el modelo relacional:** Los metadatos deben de ser accesibles usando un esquema relacional.
5. **Sublenguaje de datos completo:** Al menos debe de existir un lenguaje que permita el manejo completo de la base de datos. Este lenguaje, por lo tanto, debe permitir realizar cualquier operación.
6. **Actualización de vistas:** El DBMS debe encargarse de que las vistas muestren la última información.
7. **Inserciones, modificaciones y eliminaciones de dato nivel:** Cualquier operación de modificación debe actuar sobre conjuntos de filas, nunca deben actuar registro a registro.
8. **Independencia física:** Los datos deben de ser accesibles desde la lógica de la base de datos aún cuando se modifique el almacenamiento.
9. **Independencia lógica:** Los programas no deben verse afectados por cambios en las tablas.
10. **Independencia de integridad:** Las reglas de integridad deben almacenarse en la base de datos (en el diccionario de datos), no en los programas de aplicación.
11. **Independencia de la distribución:** El sublenguaje de datos debe permitir que sus instrucciones funciones igualmente en una base de datos distribuida que en una que no lo es.
12. **No subversión:** Si el DBMS posee un lenguaje que permite el recorrido registro a registro, éste no puede utilizarse para incumplir las reglas relacionales.

5.2.3. Base de datos No Relacionales (NoSQL)

Nace como respuesta a la necesidad de gestionar volúmenes masivos de información y que engloba todas las tecnologías de almacenamiento estructurado que no cumplen el esquema relacional [29]. En las bases NoSQL con modelos agregados, prima la velocidad, el manejo de grandes volúmenes de datos y la posibilidad de tener un sistema distribuido [30].

Las bases de datos NoSQL con modelos agregados realiza su organización de los datos de forma no normalizada, además algunas implementan uso de código abierto, lo que implica que cualquiera puede consultar su código libremente, actualizarlo según sus necesidades y compilarlo [31]. Dentro de los tipos de bases de datos NoSQL, encontramos: [29]

- **Basadas en Clave/Valor:** Donde se almacenan datos asociados a una clave. Estas son consideradas sencillas y de mayor rendimiento.
- **Basadas en Documentos:** Similar a las Key-Value, pero el valor puede ser un documento, esto hace que en nivel de complejidad aumente al realizar una consulta. Del mismo modo, también posibilitan el manejo de un lenguaje de consulta que permite realizar consultas más complejas, con agresividad similar a las SQL.
- **Basadas en Familias de Columnas:** Es una base de almacenamiento distribuido que gestiona datos estructurados y está diseñado para escalarse a tamaños muy grandes. La base está indexado por una clave de fila, una clave de columna y una fecha y hora; cada valor de la base es una matriz de bytes sin interpretar.
- **Basadas en Grafos:** Los datos se organizan en nodos y arcos. A pesar de ser catalogada como una base NoSQL, maneja una gran similitud a las bases relacionales (SQL), lo que implica que maneja restricciones al aplicarla en los criterios de manejo de grandes volúmenes de datos.

A continuación, en la figura 5.3 se ilustran algunos ejemplos de los modelos de datos de algunas bases consideradas como NoSQL[3]:

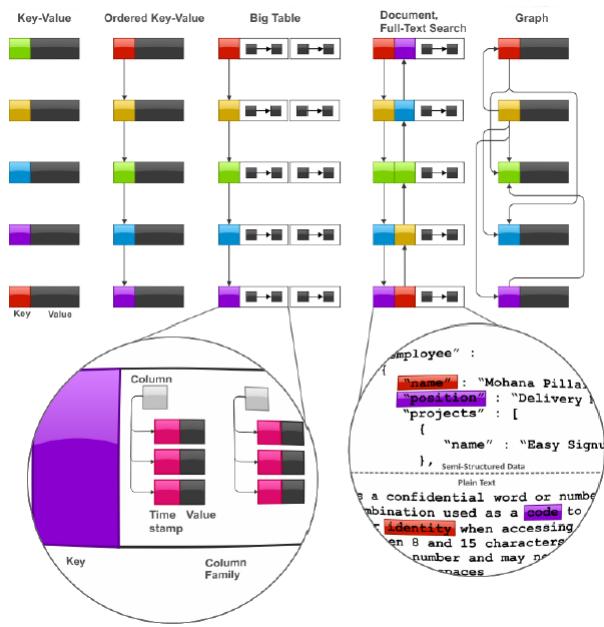


Figura 5.3: Bases de datos NoSQL[3].

5.2.3.1. Modelos que se aplican a las Bases NoSQL

1. **ACID free:** El término ACID hace referencia a la Atomicidad, Consistencia, Aislamiento y Durabilidad [31].
2. **BASE:** Sus siglas significan Basically, Available, Soft state y Eventual consistency. Es una forma distinta de manejar la transaccionalidad, que favorece el alto desempeño y perjudica la consistencia. Para buscar un alto desempeño, las bases de datos NoSQL se guían entre el camino de ACID a BASE [31].

Al igual que en las bases SQL, existen 4 componentes claves en el bloque de construcción de una base de datos NoSQL [31]:

- **Lenguaje de Modelado:** Describirá la estructura de la bases de datos y del mismo modo, define el esquema en el que se va a basar.
- **Estructura de la base de datos:** Cada base emplea su propia estructura y almacena la información dando uso de un dispositivo de almacenamiento permanente.
- **Lenguaje de consulta de base de datos:** Permite realizar todas las operaciones de creación, actualización, lectura y borrado de los datos.

- **Transacciones:** Una transacción es una herramienta que posibilita a los programadores asignar al software del sistema de base de datos la responsabilidad de evitar daños a los datos causados por amenazas como ejecuciones concurrentes, ejecuciones parciales o fallos del sistema. Durante cualquier transacción de datos, puede haber cualquier tipo de falla; por la tanto, la máquina no debe dejar de funcionar.

5.2.4. Bases de Datos Relacionales VS NoSQL

Tanto las bases relacionales como NoSQL, buscan un mismo objetivo que es el almacenamiento de datos, pero lo hacen de manera muy diferente [30].

5.2.4.1. Bases de Datos Relacionales:

- Necesidad una base de datos estructurada y organizada (la esencia de las bases de datos relacionales).
- Tipo y consistencia de datos muy importante.
- Necesidad de búsquedas complejas.
- Sencillez en la escritura.
- Necesidad recurrente de escritura y modificaciones de datos sobre elementos específicos (Bases de Datos Relacionales permiten modificar fácilmente datos específicos).

5.2.4.2. NoSQL:

- Bases de datos sin esquemas específicos.
- Versatilidad
- A partir del diseño del modelo de la base de datos, todos los datos necesarios se pueden recuperar efectivamente.
- Datos distribuidos (varias fuentes), modelos agregados como (key-value, column family, documents) implementan esta característica.

En la imagen 5.4, se muestra una comparación frente al modelo de datos y ventajas de las bases SQL y NoSQL.

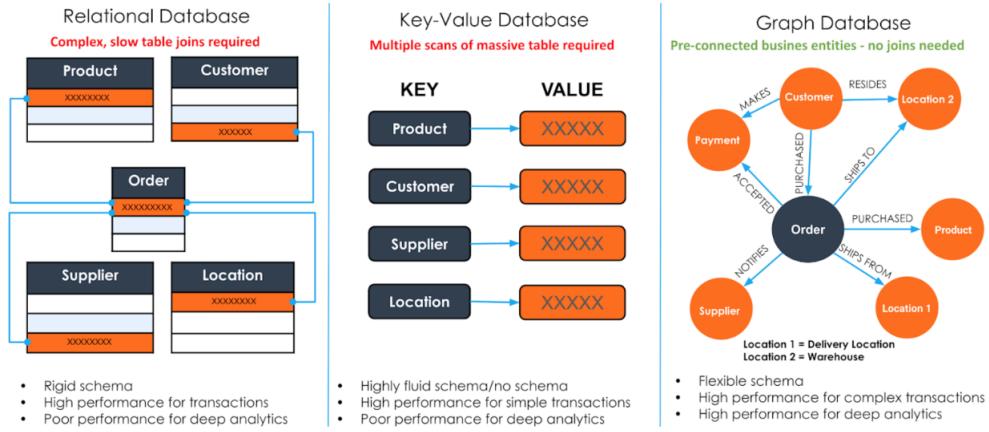


Figura 5.4: Modelo de datos y desafíos bases SQL y NoSQL. [4]

En la actualidad, las bases más implementadas son las relacionales. Estas se basan en modelos relacionales, en donde se implementa una forma intuitiva y directa de representar los datos en tablas. Pero estas bases de datos se enfrentan a diferentes desafíos en cuanto su escalabilidad y rendimiento en sus consultas, por ejemplo, las consultas SQL largas requieren más tiempo para ejecutarse. Por otro lado, las bases orientadas a grafos nos otorgan sencillez, lo que hace más simple la interpretación y la generación de consultas.

5.2.5. Base de datos orientada a grafos

Una base de datos orientada a grafos, se centra en la teoría de los grafos, donde existen un conjunto de objetos (vértices y aristas) que permiten la representación de datos interconectados. Estas bases de datos son parte de la familia de base de datos NoSQL, por ende, también pueden usarse en contextos variados y con fines muy distintos, permiten analizar información y entender, evaluar y aprovechar los procesos y las relaciones [32].

Uno de los principales lenguajes implementados para el uso de bases de datos orientadas a grafos es Cypher, un lenguaje de consulta declarativo que se basa en los conceptos básicos y cláusulas de las bases de datos SQL, pero con el plus de generar una funcionalidad adicional, facilitando la implementación de grafos. El modelo de datos que utiliza Cypher es el de grafos de propiedades, el cual comprende nodos, que representan entidades (como personas, cuentas bancarias, departamentos, etc.) y relaciones (sinónimo de bordes) representando las conexiones o relaciones entre las entidades [6].

En la imagen 5.5, se muestra un ejemplo de una base de datos orientada a grafos implementada en el programa de desarrollo Neo4j. Este desarrollo se centra en el área de la medicina, donde muestra la visualización generada a partir de informes de mamografía, ecografía y resonancia magnética de un determinado paciente [5].

La información que presentan los nodos en la imagen 5.5 son:

- Clasificación del estudio (Nodo de color morado).
- Composición o detalle de la lesión (Nodo de color azul).
- Enfermedad o lesión encontrada (Nodo de color amarillo).
- Estudio Realizado (Nodo de color rojo).
- Localización de la enfermedad (Parte del cuerpo) (Nodo de color gris).
- Lado en que se encuentra la lesión o enfermedad (Nodo de color verde).

Del mismo modo, las relaciones que existen entre dichos nodos está dada por las palabras claves: **tiene, presenta, clasificada como, localizada en, e incluye**.

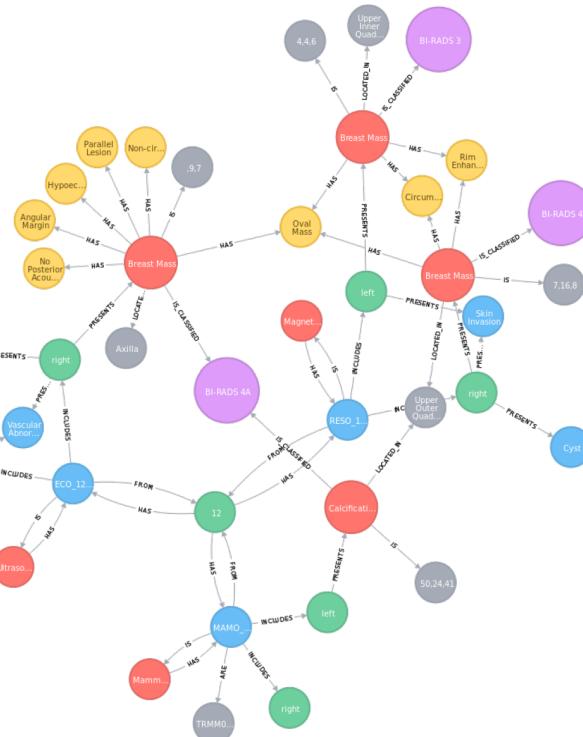


Figura 5.5: Base de datos de grafos, ilustración de una red de informes médicos. [5]

5.2.5.1. Neo4j

Es un sistema gestor para la creación de bases de datos. Neo4j permite almacenar el grafo de forma nativa en el disco y proporciona un marco para recorrer y ejecutar operaciones en base a grafos. Por lo tanto, el lenguaje se usa en cientos de aplicaciones de producción en muchos dominios verticales de la industria [6].

5.2.5.2. Lenguaje de consulta Cypher y OpenCypher

El lenguaje de consulta de grafos de propiedades Cypher es un lenguaje en evolución, originalmente diseñado e implementado como parte de la base de datos de grafos Neo4j, actualmente es utilizado por varios productos de bases de datos comerciales e investigadores [6].

Una consulta se encuentra compuesta por varias cláusulas encadenadas. Cada cláusula maneja como entrada el estado del grafo y una tabla de resultados intermedios formada por las variables actuales, y esta genera una salida, que es un nuevo estado del grafo y una nueva tabla de resultados intermedios. Una ventaja del lenguaje Cypher es que sólo materializa resultados intermedios cuando es necesario [33].

El modelo de datos en Neo4j, gestionado por el lenguaje de consulta Cypher, se basa en grafos de propiedades. Este modelo es fundamental en la representación de entidades y relaciones. Los nodos, que actúan como entidades, pueden representar diversos elementos, como personas, cuentas bancarias o departamentos, mientras que las relaciones, también conocidas como bordes, simbolizan las conexiones entre estas entidades [6].

A continuación, nos enfocaremos en las capacidades de este lenguaje, desde su capacidad para realizar consultas lineales, pasando por su eficiente capacidad de modificar datos hasta su enfoque pragmático al proporcionar soporte para parámetros de consulta. Además, exploraremos su utilidad en la coincidencia de patrones, presentando una manera efectiva y visual de consultar bases de datos.

- **Consultas lineales:** Una consulta Cypher toma como entrada un grafo de propiedades y genera una tabla. Cada cláusula en una consulta es una función que toma una tabla y genera una nueva que puede expandir el número de campos y agregar nuevas tuplas. Es decir, una consulta es planteada a partir de la composición de funciones.

El flujo lineal de consultas en Cypher se extiende a la composición de consultas. El **WITH** cláusula permite las mismas proyecciones que **RETURN**, incluidas las agregaciones. Además de esta forma lineal de componer consultas, Cypher también admite subconsultas anidadas como **UNION** [6].

- **Modificación de datos:** Cypher admite un lenguaje de actualización enriquecido para la modificación de la base de datos. Las cláusulas de actualización reutilizan el lenguaje de pa-

trón gráfico visual y proporcionan el mismo modelo semántico simple.

Por ejemplo, para crear un nuevo nodo con una etiqueta y una propiedad, se puede usar la palabra clave **CREATE** seguida del patrón del nodo. Para asegurar que no se creen nodos duplicados, se puede usar la palabra clave **MERGE**, que busca si existe el nodo antes de crearlo [6].

- **Pragmático:** Cypher es intencionalmente similar a SQL para ayudar a los usuarios a realizar la transición entre los dos idiomas. Sigue la misma estructura de sintaxis de cláusulas e implementa la semántica establecida para muchas funciones. Cypher tiene soporte incorporado para los parámetros de consulta, lo que facilita la eliminación de los problemas de inyección de consultas [6].
- **La coincidencia de patrones:** es una forma de consultar una base de datos que usa lenguaje cypher. Los patrones en Cypher se expresan de forma visual como ASCII, que son símbolos que representan los nodos y las relaciones de un grafo. Por ejemplo, el patrón “(a)-[r]->(b)” significa que hay un nodo “a” que tiene una relación “r” con otro nodo “b”. Los patrones pueden tener propiedades, etiquetas y variables para especificar las condiciones de la consulta [6].
- **Implementación de Neo4j:** La ejecución de consultas en Neo4j sigue un modelo convencional, descrito por Volcano Optimizer Generator. La planificación de consultas en Neo4j se basa en el algoritmo IDP, usando un modelo de costos. La compilación final de la consulta utiliza un modelo de ejecución simple basado en un iterador tupla a la vez, o compila la consulta en código de bytes de Java con un modelo de ejecución basado en push [6].

A continuación, en la figura 5.6 se muestra una base de datos G formado por investigadores, estudiantes y publicaciones. Para cada investigador, se muestran los estudiantes que supervisan y las publicaciones que han escrito, y para cada publicación, mostramos qué otras publicaciones cita [6]. La información que presentan los nodos en la imagen 5.6 son:

- Investigadores (Nodo de color morado).
- Estudiantes (Nodo de color rojo).
- Publicaciones (Nodo de color amarillo).

Del mismo modo, las relaciones que existen entre dichos nodos está dada por las palabras claves: autores, ciudades y supervisores. Es fundamental comprender que en este contexto, los nodos se representan mediante círculos, mientras que los cuadros representan los datos asociados a cada nodo y a sus arcos, los cuales están compuestos por etiquetas y atributos.

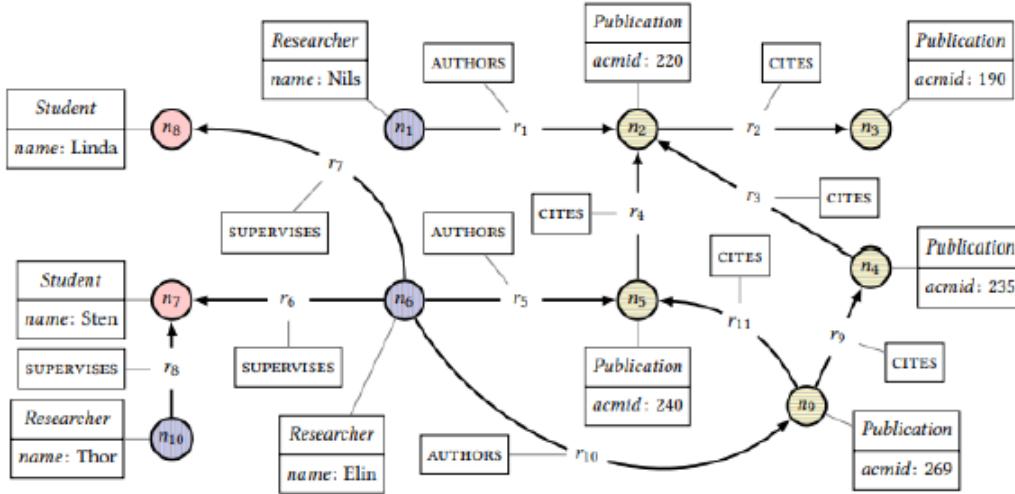


Figura 5.6: Base de datos de referencia que muestra la información de supervisión y citas para investigadores, estudiantes y publicaciones [6]

A partir de esto, se dará un ejemplo del uso de las consultas en el lenguaje de programación Cypher. La siguiente consulta 5.7 devuelve el nombre de cada investigador en G, la cantidad de estudiantes que supervisan actualmente y la cantidad de veces que una publicación de la que son autores ha sido citada, tanto directa como indirectamente, por otras publicaciones.

```

1 MATCH (r:Researcher)
2 OPTIONAL MATCH (r)-[:SUPERVISES]->(s:Student)
3 WITH r, count(s) AS studentsSupervised
4 MATCH (r)-[:AUTHORS]->(p1:Publication)
5 OPTIONAL MATCH (p1)<-[:CITES*]-(p2:Publication)
6 RETURN r.name, studentsSupervised,
7      count(DISTINCT p2) AS citedCount

```

Figura 5.7: Consulta en Cypher [6]

r	s	r	studentsSupervised
n ₁	null	n ₁	0
n ₆	n ₇	n ₆	2
n ₆	n ₈	n ₁₀	1
n ₁₀	n ₇		

(a)

(b)

Figura 5.8: Resultado de línea 2 (a), línea 3 (b)[6]

El patrón dado en la primera cláusula **MATCH** de la línea 1 coincide con todos los investiga-

dores; es decir, los nodos con la etiqueta Investigador. Esto produce tres vínculos para la variable r, a saber, n1, n6 y n10 representados como tres filas en una tabla con un único atributo r, que corresponden a los investigadores Nils, Elin y Thor [6].

La cláusula **MATCH** de la línea 1, tiene una variante opcional: **OPTIONAL MATCH**, que es análoga a la construcción outer join de SQL. Esta cláusula produce filas para todas las coincidencias del mismo modo que **MATCH**, siempre que se encuentre el patrón completo en la base de datos. Sin embargo, en los casos en que no se encuentre ningún dato que coincida con todo el patrón, se producirá una única fila en la que los enlaces de todas las variables introducidas en **OPTIONAL MATCH** se establecerán en null.

La cláusula **OPTIONAL MATCH** de la línea 2 empareja a todos los estudiantes supervisados por cada investigador. Esto produce un enlace de la variable recién introducidas para cada valor al que r estaba enlazado por la cláusula **MATCH** de la línea 1. Cuando r está vinculada a n1 (Nils, que no supervisa a ningún estudiante), la vinculación correspondiente para s es nula. En la figura 5.8 (a), muestra el resultado de este **OPTIONAL MATCH**.

La cláusula **WITH** de la línea 3 se utiliza tanto para proyectar un subconjunto de las variables actualmente en el ámbito y sus ligaduras a la parte de la consulta que sigue a **WITH**, como para calcular una agregación. La cláusula **WITH** tiene dos expresiones, la segunda de las cuales es una agregación que funciona de forma muy similar a SQL. La primera expresión, r, es una expresión no agregadora y, por lo tanto, actúa como una clave de agrupación implícita para la función agregadora count(s). Tomando los resultados en la figura 5.8 (b), contamos todos los valores no nulos de s para cada enlace único de r, con el alias estudiantesSupervisado.

La cláusula **WITH** proyectará todas las vinculaciones producidas para r y estudiantesSupervisado. Observamos que la variable s ya no está en el ámbito después de la línea 3, ya que no fue proyectada por **WITH** y ya no puede utilizarse en el resto de la consulta. Esta tabla actúa ahora como tabla impulsora de la cláusula **MATCH** de la línea 4. Las variables de enlace producidas por la línea 4 incluyen a todos los investigadores que son autores de al menos una publicación, el número de estudiantes que supervisan y la publicación de la que son autores. El resultado se puede apreciar en la figura 5.9.

r	studentsSupervised	p1
n1	0	n2
n6	2	n5
n6	2	n9

Figura 5.9: Resultado de línea 2 - 4 [6]

La cláusula **OPTIONAL MATCH** de la línea 5 empareja, para cada publicación de la que es autor uno de los investigadores de G , todas las publicaciones que lo citan, tanto directa como indirectamente. Esto se consigue mediante el uso de un patrón que contiene la relación de longitud variable CITES, que indica que deben recorrerse una o varias relaciones CITES. Los resultados obtenidos se visualizan en la figura 5.10.

r	studentsSupervised	p1	p2
n ₁	0	n ₂	n ₄
n ₁	0	n ₂	n ₉ †
n ₁	0	n ₂	n ₅
n ₁	0	n ₂	n ₉ †
n ₆	2	n ₅	n ₉
n ₆	2	n ₉	null

Figura 5.10: Resultado de línea 5[6]

Cuando p1 está ligado a n₉, que no es citado por ninguna publicación, la ligadura correspondiente para p2 es nula. Además, observamos que hay dos filas idénticas, indicadas con †. La existencia de estas filas duplicadas se debe al patrón de relación de longitud variable: n₉ es alcanzable desde n₂ a través de los nodos intermedios n₅ y n₄.

RETURN es la última cláusula de la consulta (líneas 6 y 7), que a diferencia de las otras consultas, esta no genera resultados intermedios, si no que es el resultado final al conjunto de consultas generadas. Se proyecta el valor de la propiedad name de cada investigador, junto con el valor de studentsSupervised. La expresión de agregación, a diferencia de la de la línea 3, cuenta los valores distintos de p2 (excluyendo los valores nulos) y aliasas los resultados como citedCount, que denota el número de publicaciones que citan una publicación cuyo autor es el investigador. La tabla formada por las filas que contienen el resultado de las expresiones se proyecta mediante **RETURN** en la figura 5.11:

r.name	studentsSupervised	citedCount
Nils	0	3
Elin	2	1

Figura 5.11: Resultado de línea 6 y 7 [6]

5.3. Protocolos de Comunicación Industriales

Los protocolos de comunicación forman parte de nuestra investigación, para el entendimiento de las comunicaciones que se presentan en la funcionalidad de la celda de Manufactura Flexible del CAP. Esto con el fin de conocer su funcionamiento a profundidad y de los protocolos y normas existentes detrás de estas comunicaciones.

5.3.1. OPC UA (Arquitectura Unificada de Comunicaciones de Plataforma Abierta)

Es una arquitectura orientada a servicios independiente de la plataforma que integra toda la funcionalidad de las especificaciones individuales de OPC Classic en un marco extensible que proporciona soluciones más robustas, seguras y escalables para la comunicación industrial [34] [35].

A diferencia de OPC Classic, OPC UA puede funcionar en diferentes sistemas operativos, como Windows, Linux y otros. Utiliza protocolos estándar de Internet, como TCP/IP y HTTPS, para la comunicación, lo que facilita la interoperabilidad entre dispositivos y aplicaciones en entornos heterogéneos. Una de las principales ventajas de OPC UA es su enfoque en la seguridad. Proporciona mecanismos de autenticación, autorización y cifrado de extremo a extremo para proteger la comunicación y los datos transmitidos. Además, ofrece características avanzadas, como un modelo de información flexible, soporte para datos históricos, notificación de eventos y gestión de alarmas [34] [36] [37].

OPC-UA se ha convertido en el estándar de facto para la comunicación industrial y ha sido adoptado por muchas empresas y organizaciones en todo el mundo. La norma **IEC 62541** define las especificaciones técnicas para la implementación de OPC UA, brindando un marco común y coherente para su implementación y uso en diversos sectores de la industria [38] [39].

Las principales especificaciones técnicas definidas por la norma **IEC 62541** son las siguientes:

1. **Modelo de información:** Se establecen las estructuras de datos y la semántica utilizada para describir la información intercambiada entre los sistemas a través de OPC UA. Esto incluye la definición de nodos, atributos, métodos y eventos.
2. **Protocolo de comunicación:** Se especifica el protocolo de comunicación utilizado para la transferencia de datos entre los sistemas. OPC UA utiliza un modelo cliente-servidor, donde los clientes solicitan y reciben datos de los servidores. El protocolo puede basarse en TCP/IP y soportar diferentes mecanismos de transporte y seguridad.
3. **Seguridad:** Se definen los mecanismos de seguridad utilizados para proteger la comunicación y los datos transmitidos en OPC UA. Esto incluye autenticación, autorización, cifrado y firma digital, entre otros aspectos.

4. **Descubrimiento y conexión:** Se describe cómo los clientes pueden descubrir y conectarse a los servidores OPC UA disponibles en la red. Esto puede implicar la resolución de nombres, la búsqueda de servicios de directorio y la negociación de las opciones de comunicación.
5. **Historial de datos:** Se establece cómo se almacenan y acceden a los datos históricos en OPC UA. Esto puede incluir la capacidad de almacenamiento local en los servidores y la consulta de datos históricos por parte de los clientes.
6. **Alarms y eventos:** Se establece el mecanismo para el manejo de alarmas y eventos en OPC UA. Permite a los servidores notificar a los clientes sobre cambios de estado, condiciones de error o eventos relevantes.

5.3.2. Ethernet Industrial

Es el uso de Ethernet en un entorno industrial con protocolos que proporcionan determinismo y control en tiempo real. Los protocolos para Ethernet industrial incluyen EtherCAT , EtherNet/IP , PROFINET , POWERLINK , SERCOS III , CC-Link IE y Modbus TCP [40]. En el caso de la celda de manufactura del CAP, esta utiliza el protocolo de Modbus TCP (también Modbus TCP/IP), el cual es una extensión de Modbus RTU con una interfaz TCP que se ejecuta en Ethernet a través del puerto 502 y fue desarrollado originalmente por Schneider Electric. Además, Modbus es un bus serial sencillo, robusto y libre de derechos de autor que es fácil de implementar y funciona en enlaces físicos RS-232 o RS-485 con velocidades de hasta 115K baudios. [41] [42].

Por otro lado, Ethernet se está convirtiendo en omnipresente y rentable, con enlaces físicos comunes y mayor velocidad, por ello, muchos protocolos de comunicación industrial se están pasando a soluciones basadas en esta tecnología, permitiendo así una topología de red flexible y un número de nodos en el sistema [40].

5.3.3. Modbus TCP/IP

Fue desarrollado bajo la unión del protocolo de aplicación Modbus con la transmisión Ethernet **IEEE 802.3** tradicional. El Protocolo de control de transporte (TCP) reside una capa por encima del Protocolo de Internet (IP) y es responsable de transportar los datos de la aplicación y hacerlos seguros, mientras que IP es responsable del direccionamiento real y la entrega de los datos [43].

Por un lado, el paquete TCP se inserta en la porción de datos del paquete IP debajo de él, mientras que IP en sí mismo es un protocolo no seguro y sin conexión y debe funcionar junto con el TCP superpuesto para poder operar. De esta forma, TCP generalmente se considera la capa superior de la plataforma IP que sirve para garantizar la transferencia segura de datos [41]. Para implementar una comunicación Modbus se deben tener en cuenta las siguientes reglas según la organización de Modbus [44]:

1. Sin requisitos explícitos del usuario, se recomienda implementar la gestión automática de conexiones TCP.

2. Se recomienda mantener la conexión TCP abierta con un dispositivo remoto y no abrirla y cerrarla para cada transacción MODBUS/TCP. Observación: Sin embargo, el cliente MODBUS debe ser capaz de aceptar una solicitud de cierre del servidor y cerrar la conexión. . La conexión se puede reabrir cuando sea necesario.
3. Se recomienda que un Cliente MODBUS abra un mínimo de conexiones TCP con un servidor MODBUS remoto (con la misma dirección IP). Una conexión por aplicación podría ser una buena opción.
4. Se pueden activar varias transacciones MODBUS simultáneamente en la misma conexión TCP. Observación: si se hace esto, entonces se debe usar el identificador de transacción MODBUS para identificar de manera única las solicitudes y respuestas coincidentes.
5. En el caso de una comunicación bidireccional entre dos entidades MODBUS remotas (cada una de ellas es cliente y servidor), es necesario abrir conexiones separadas para el flujo de datos del cliente y para el flujo de datos del servidor.
6. Una trama/frame TCP debe transportar solo una ADU MODBUS. Se desaconseja enviar múltiples solicitudes o respuestas MODBUS en la misma PDU TCP.

5.3.4. Comunicación Serial RS232 y señales digitales I/O

El protocolo de comunicación RS232 (Estándar Recomendado 232 por sus siglas en inglés) fue introducido por primera vez en 1960 por la Electronic Industries Association (EIA) es uno de los más antiguos pero populares que se utiliza a nivel industrial. Este es un tipo de tipo de comunicación serial utilizada para la transmisión de datos normalmente en distancias medias [45]. No obstante, el estándar ha cambiado de nombre varias veces durante su historia a medida que la organización cambió su nombre, y ha sido conocido como EIA RS-232, EIA 232 y, más recientemente, como TIA 232 desde 1988 por la Telecommunications Industry Association (TIA).

Debido a que RS-232 se usa más allá del propósito original de interconectar un terminal con un módem, se han desarrollado estándares sucesores para abordar las limitaciones. Los problemas con el estándar RS-232 incluyen [46]:

- Las grandes oscilaciones de voltaje y el requisito de suministros positivos y negativos aumentan el consumo de energía de la interfaz y complican el diseño de la fuente de alimentación. El requisito de oscilación de voltaje también limita la velocidad superior de una interfaz compatible.
- La señalización de un solo extremo referida a una tierra de señal común limita la inmunidad al ruido y la distancia de transmisión.
- La conexión multipunto entre más de dos dispositivos no está definida. Si bien se han ideado "soluciones alternativas" multipunto, tienen limitaciones en cuanto a velocidad y compatibilidad.

- El estándar no aborda la posibilidad de conectar un DTE directamente a un DTE, o un DCE a un DCE. Se pueden usar cables de módem nulo para lograr estas conexiones, pero estos no están definidos por el estándar, y algunos de estos cables usan conexiones diferentes que otros.
- No se especifica ningún método para enviar energía a un dispositivo. Si bien se puede extraer una pequeña cantidad de corriente de las líneas DTR y RTS, esto solo es adecuado para dispositivos de bajo consumo, como ratones .
- El conector D-sub de 25 pines recomendado en el estándar es grande en comparación con la práctica actual.

En cuanto a las máquinas de la celda de manufactura del CAP, los controladores de dispositivo tanto para los tornos como para las fresadoras Denford se comunican con las herramientas de máquina de dos maneras diferentes: Enlaces seriales RS232 y señales digitales de Entrada y Salida (I/O) [47].

En el caso de RS232 se utiliza para la descarga de programas: control numérico directo (DNC) y verificación detallada del estado. Se debe conectar un cable de RS232 desde un puerto serial en la PC de la celda supervisora al puerto serial externo en la máquina herramienta. Para comunicarse a través del enlace serial, tanto la máquina herramienta como el controlador del dispositivo deben usar la misma configuración [47].

Las señales I/O se conectan directamente desde el módulo de interfaz al puerto de entrada y salida auxiliar de la máquina. Hay dos líneas de entrada y dos líneas de salida que se pueden usar para transmitir información de estado de bajo nivel al controlador del dispositivo. Todas las herramientas Denford están conectadas al módulo de interfaz de I/O en los puertos 2A o 2B [47]. El puerto auxiliar proporciona dos entradas a la herramienta, dos salidas de la máquina y un circuito de parada de emergencia. Las entradas y salidas hacia y desde la herramienta corresponden a cuatro códigos 'M':

- M62: salida 1 activada.
- M63: salida 2 activada.
- M64: salida 1 desactivada
- M65: salida 2 desactivada

5.4. Estado del Arte

Para la realización de este trabajo se consultaron diferentes trabajos enfocados en el mejoramiento de los procesos de control y la gestión de datos en entornos industriales mediante bases de datos orientadas a grafos. Estos trabajos se resumen a continuación:

5.4.1. Wireless cyber-physical systems performance evaluation through a graph database approach

En este trabajo, los autores se centran en la evaluación del rendimiento de los sistemas inalámbricos ciberfísicos utilizando un enfoque de base de datos de grafos para modelar y analizar los sistemas ciberfísicos, combinando elementos físicos y de software en una red inalámbrica. El artículo proporciona una visión general del enfoque utilizado, destacando la importancia de evaluar el rendimiento de estos sistemas para garantizar su eficiencia y fiabilidad. Los autores presentan los resultados obtenidos a través del análisis de rendimiento realizado utilizando la base de datos de grafos. Estos resultados proporcionan información sobre aspectos como la latencia, el ancho de banda y la calidad de servicio del sistema evaluado, mediante el cálculo de métricas de rendimiento de la red y descubrir correlaciones [13].

5.4.2. Lenguaje visual de consulta basado en transformación de grafos: aplicación en el dominio médico

En este trabajo, la autora propone un lenguaje visual de consulta denominado GraphTQL, el cual se basa en un modelo de grafos. La principal premisa de GraphTQL radica en su enfoque centrado en el usuario final, buscando proporcionar una mayor expresividad en comparación con las herramientas convencionales de exploración de grafos. A diferencia de estas últimas, GraphTQL no traduce simplemente elementos de un lenguaje basado en texto a una notación visual, sino que se esfuerza por mantener la capacidad de expresión mientras se presenta de manera gráfica.

Los resultados obtenidos evidencian que GraphTQL simplifica la formulación de consultas ad hoc (no conocidas con anticipación), que pueden llegar a ser complejas. Por ende, el lenguaje GraphTQL demuestra ser una opción altamente viable en comparación con otros sistemas de bases de datos como SPARQL [11].

5.4.3. Topology Modeling and Analysis of a Power Grid Network Using a Graph Database

En este artículo, se presenta un nuevo método de arquitectura para almacenar, modelar y analizar datos de la red eléctrica. En primer lugar, los autores exponen una arquitectura para la construcción del modelo de red haciendo uso de la base de datos basada en grafos de Neo4j. Por otro lado, se presenta el diseño de los subprocesos usados para realizar el análisis de la energización inicial, el cálculo del camino más corto en conjuntos de datos pequeño a grandes y la búsqueda condicional,

para finalmente, comparar la funcionalidad y la eficiencia del modelo de base de datos propuesto y el análisis de topología contra una base de datos relacional tradicional (PostgreSQL). Los autores concluyen que el análisis de redes de energía utilizando Neo4j tiene mejor rendimiento y que para más de 20 millones de nodos, el uso de múltiples subprocessos es mucho más rápido que el uso de un solo subprocesso para el análisis inicial de energización, al igual que para el cálculo del camino más corto funciona mejor en Neo4j [48].

5.4.4. Bases de datos orientadas a grafos aplicadas al estudio de informes radiológicos: utilizando entornos de computación en la nube para abordar estudios de gran dimensión

Este trabajo, se enfoca en el uso de bases de datos orientadas a grafos para el diagnóstico y el tratamiento de informes radiológicos, específicamente del cáncer de mama. La solución propuesta se basa en la utilización de bases de datos orientadas a grafos y técnicas de procesamiento distribuido para abordar estudios de gran dimensión utilizando entornos de computación en la nube. La autora concluye que su aplicación del modelo de grafos es capaz de manejar grandes volúmenes de datos y aumentar la eficiencia del procesamiento, al igual que mejorar la representación de los informes clínicos los cuales atribuyen al desarrollo de nuevos procedimientos para el análisis y el estudio de los mismos [5].

5.4.5. A knowledge graph-based data representation approach for IIoT-enabled cognitive manufacturing

En este artículo los autores proponen un enfoque de representación de datos basado en grafos de conocimiento para la fabricación cognitiva habilitada por IIoT. El enfoque se basa en la construcción de un grafo de conocimiento de fabricación de múltiples capas que permite la toma de decisiones cognitivas desde una perspectiva global, haciendo uso de una base de datos basada en grafos, específicamente la base de datos Neo4j. Los autores definen y almacenan nodos y bordes en la base de datos para representar entidades y relaciones en la fabricación. Los nodos representan entidades como piezas, ensamblajes, características, pasos, procesos, requisitos, dispositivos y herramientas, mientras que los bordes representan relaciones como “tiene”, “tiene parte”, “parte de”, “tiene ensamblaje”, “subensamblaje de” y “orden”.

Además, los autores utilizan el modelo de datos RDF para realizar análisis semánticos y procesos de razonamiento cognitivo habilitados para IIoT en el grafo de conocimiento construido. El enfoque de grafos de conocimiento permite la fusión automática de datos heterogéneos de múltiples fuentes y la realización de razonamiento y toma de decisiones basados en el método de incrustación de grafos [49].

CAPÍTULO 6

Definición de Requisitos

6.1. Descripción del Caso de Estudio

6.1.1. Estado actual del sistema

La celda de manufactura flexible (FMC por sus siglas en inglés) del CAP es considerada como un PYME (Pequeña y Mediana Empresa), de modo que se utiliza para la creación de productos personalizados sin intervención humana durante la fabricación. Sin embargo, la celda actual del CAP necesita de la intervención de técnicos a cargo del diseño CAD/CAM, mantenimiento de máquinas, control de calidad, preparación de materias primas y envío de productos terminados, pero manteniéndose en el escenario académico sin inducir riesgos económicos y laborales. En la figura 6.1 se aprecia la celda de manufactura que se encuentra en el CAP:



Figura 6.1: FMC del CAP

La composición de la FMC del CAP está dada por las siguientes máquinas:

1. **Sistema automatizado de almacenamiento y recuperación (ASRS):** Este sistema hace el papel de almacén, en el cual se encuentran los materiales en bruto y las piezas acabadas que se realizaron a lo largo de la celda de manufactura. En la figura 6.2, se ilustra el ASRS presente en el CAP:



Figura 6.2: Almacén ASRS

Este almacén se encuentra compuesto de un robot cartesiano interno, que a partir de una serie de comandos, ejecuta la salida o entrada de material. Cabe aclarar, que actualmente la sección de la derecha de la máquina, se encuentra destinada a la materia prima, y la sección del otro lado, corresponde a todos los productos ya obtenidos al final del proceso.

2. **Estación de Torno:** Es una máquina que toma una barra de un material y hace que gire mientras una herramienta de corte va eliminando o moldeando el material de la barra hasta que quede el producto deseado. El material está asegurado y rotado por el husillo principal, mientras que la herramienta de corte se puede mover a lo largo de múltiples ejes. Los tipos de piezas creadas por un torno CNC suelen ser cilíndricas o simétricas alrededor de un eje.

Esta estación se encuentra compuesta por la máquina CNC torneadora, y un robot Mitsubishi que permite ingresar o sacar la pieza en el torno. A continuación en la figura 6.3, se mostrará la estación actual:



Figura 6.3: Estación de torneado

3. **Estación de Fresado:** Se compone de una máquina de fresado CNC la cual se utiliza para mecanizar, cortar y producir piezas diseñadas a medida en una amplia gama de materiales, como aluminio, hierro, aleaciones de acero, níquel, etc. Se le llama fresado, debido a que se realiza con una herramienta denominada “fresa” en forma de cuchillas.

En adición, también se compone por un robot Mitsubishi que permite ingresar o sacar la pieza en el torno. A continuación en la figura 6.4, se mostrará la estación actual:



Figura 6.4: Estación de Fresado

4. **Estación de Montaje y/o de Inspección:** Se compone de un robot colaborativo UR3, el cual se utiliza para la manipulación de las piezas terminadas y a su vez, para hacer el proceso de chequeo, en donde a través de una cámara el UR3 inspecciona la pieza para validar su acabado final y finalmente devolverla a la banda transportadora. En la figura 6.5, se puede apreciar la estación de Montaje actual:



Figura 6.5: Estación de Montaje

Por otro lado, la celda de manufactura del CAP además de contar con 4 estaciones, también se encuentra equipada de una banda transportadora para el transporte del material y de sensores que permiten la activación de pistones en cada estación.

Actualmente la celda de manufactura del CAP opera mediante un software de control denominado "Coordinador", que permite a través de comunicaciones como Ethernet, RS232, Modbus TCP/IP y OPC UA, activar y desactivar las estaciones. Cabe aclarar que actualmente no se cuenta con un software de control 100 % funcional, debido a que es necesario contar con un operario que especifique qué máquinas se van a activar, la ubicación exacta del material y las especificaciones técnicas que se deben de enviar para que la máquina funcione.

Del mismo modo, se resalta que la estación ASRS no cuenta con un sistema inteligente que le permita la validación del material que está buscando y de reportar que en la ubicación dada no se encuentra el material solicitado.

6.1.2. Bloques de Conexión

Actualmente, la FMC del CAP implementa protocolos de comunicación Ethernet, OPC UA y Serial para el control y funcionamiento del proceso. Este control es realizado por un **coordinador** en este caso, un software de control, que mediante un programa controla la lógica y el funcionamiento de la celda.

6.1.2.1. Diagrama Global

A continuación, en la figura 6.6 se ilustra el diagrama global de los bloques de conexión de la Celda de Manufactura del CAP:

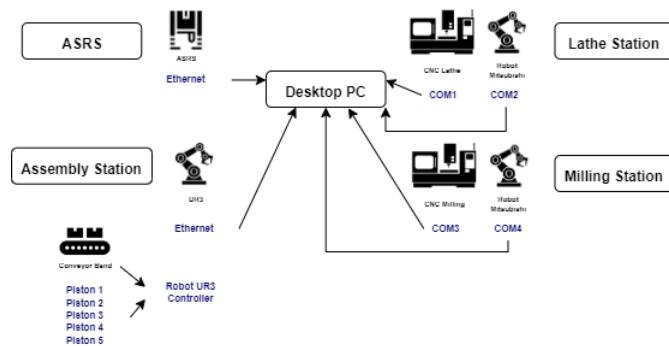


Figura 6.6: Diagrama Global de la Celda de Manufactura del CAP

6.1.2.2. Diagrama de comunicación Ethernet

Tanto el almacén ASRS y el robot UR3 cuentan con la facilidad de conectarse por medio de Ethernet, permitiendo disminuir la conexión serial. En la figura 6.7, se ilustra la comunicación Ethernet presente en la estación ASRS y el robot UR3.

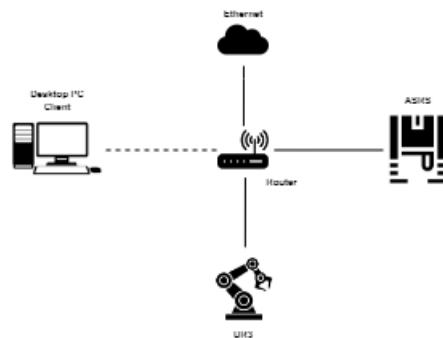


Figura 6.7: Diagrama de Comunicación Ethernet Celda de Manufactura del CAP

6.1.2.3. Diagrama de Comunicación OPC UA

Este tipo de comunicación fue implementada recientemente, con el fin de conseguir una ejecución en paralelo, es decir, poder realizar una comunicación entre cada estación simultáneamente. Esta comunicación se ilustra a continuación en la figura 6.8:

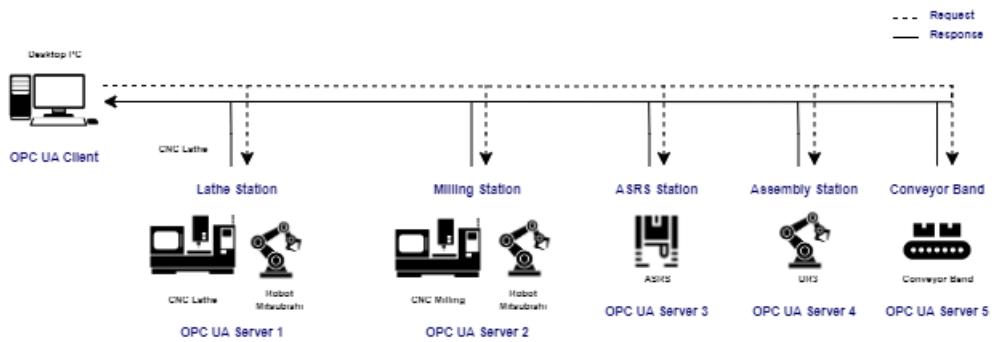


Figura 6.8: Diagrama de comunicación OPC UA Celda de Manufactura del CAP

6.1.3. Lógica Funcional

A partir de las visitas y reuniones con los expertos del CAP, se determinó que la celda de manufactura flexible presente, consta de los siguientes pasos para el desarrollo de una orden:

1. Creación de Orden (Material,Cantidad y Pieza).
2. Validación de disponibilidad de material: Este proceso es realizado mediante un operario, donde especifica la ubicación exacta del material después de haber validado que ahí se encuentre disponible.
3. ASRS trae el material de la sección de materia prima y es colocado en la banda transportadora: Como se mencionó anteriormente, el ASRS va a la ubicación brindada y trae la paleta correspondiente, más no valida que haya material en ella o que el material corresponda al indicado.
4. Mediante un script de Python, que debe de ser ejecutado por el operario, se realizan los siguientes pasos a continuación:
5. Se activa la banda transportadora.
6. Funcionamiento Estación de Torno.
7. Funcionamiento Estación de Fresado.
8. Funcionamiento Estación de Montaje.

9. Funcionamiento ASRS acomodar pieza en la sección de terminados.
10. Mediante un script de Python, se alimenta una base de datos SQL para el registro de los datos obtenidos en la ejecución de la celda. Este script es ejecutado por el operario.

Cabe aclarar, que el orden de los pasos descritos puede verse afectado por las características de la pieza a realizar. Es decir, pueden existir piezas que para su fabricación no tengan la necesidad de pasar por todas las estaciones.

En adición, un detalle relevante en la definición de la lógica funcional, es conocer que la estructura en la que se encuentra posicionada la celda de manufactura y sus máquinas, no permiten gestionar varios pedidos simultáneamente, es decir, se ejecuta orden por orden, y como cada orden cuenta con una cantidad a producir, del mismo modo, la producción de piezas dependerán de la disponibilidad de la máquina. A partir de esto, en las figuras 6.9 y 6.10 se puede ver la lógica funcional completa de la celda de manufactura del CAP:

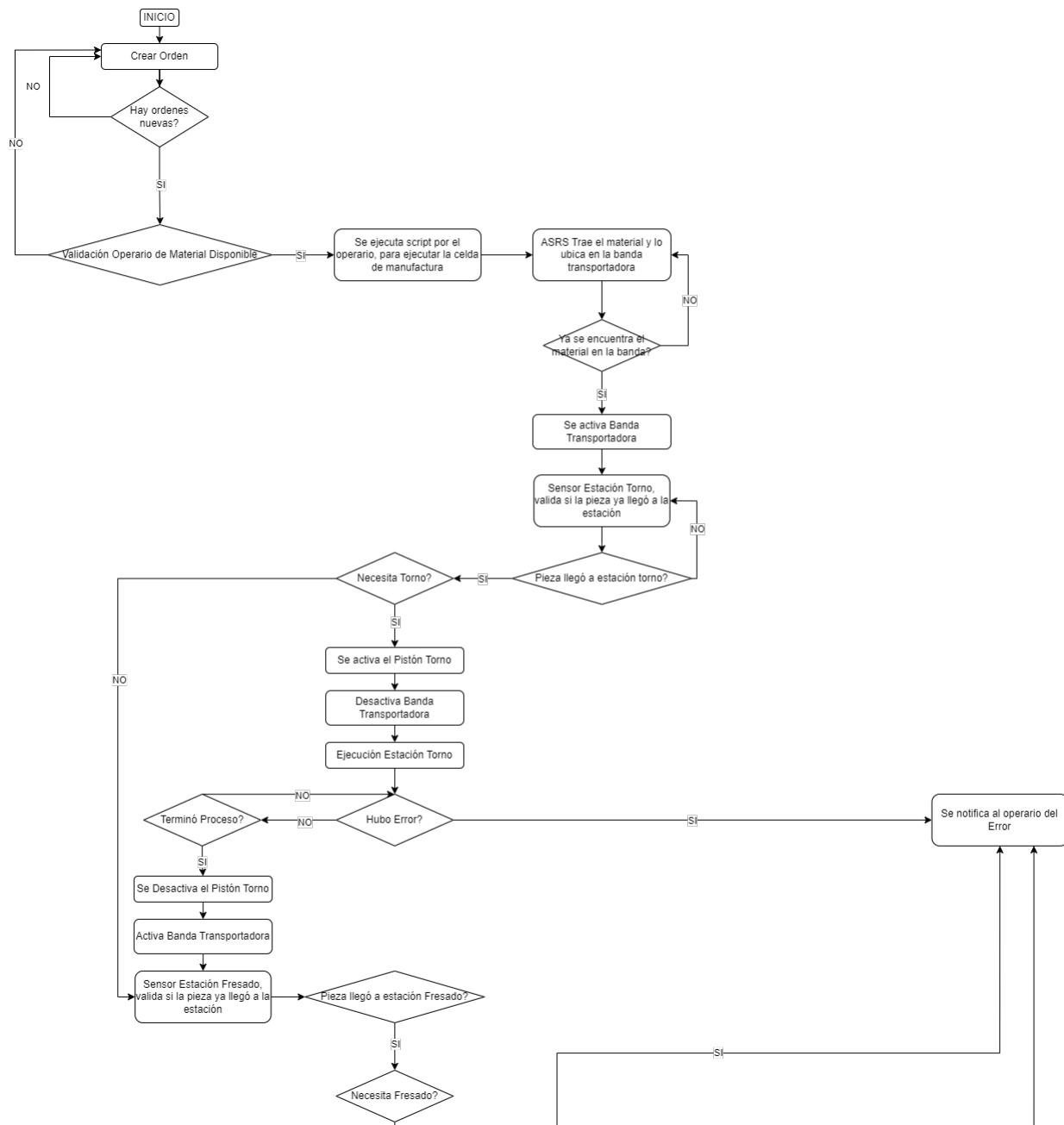


Figura 6.9: Lógica Funcional Celda de manufactura CAP Parte 1

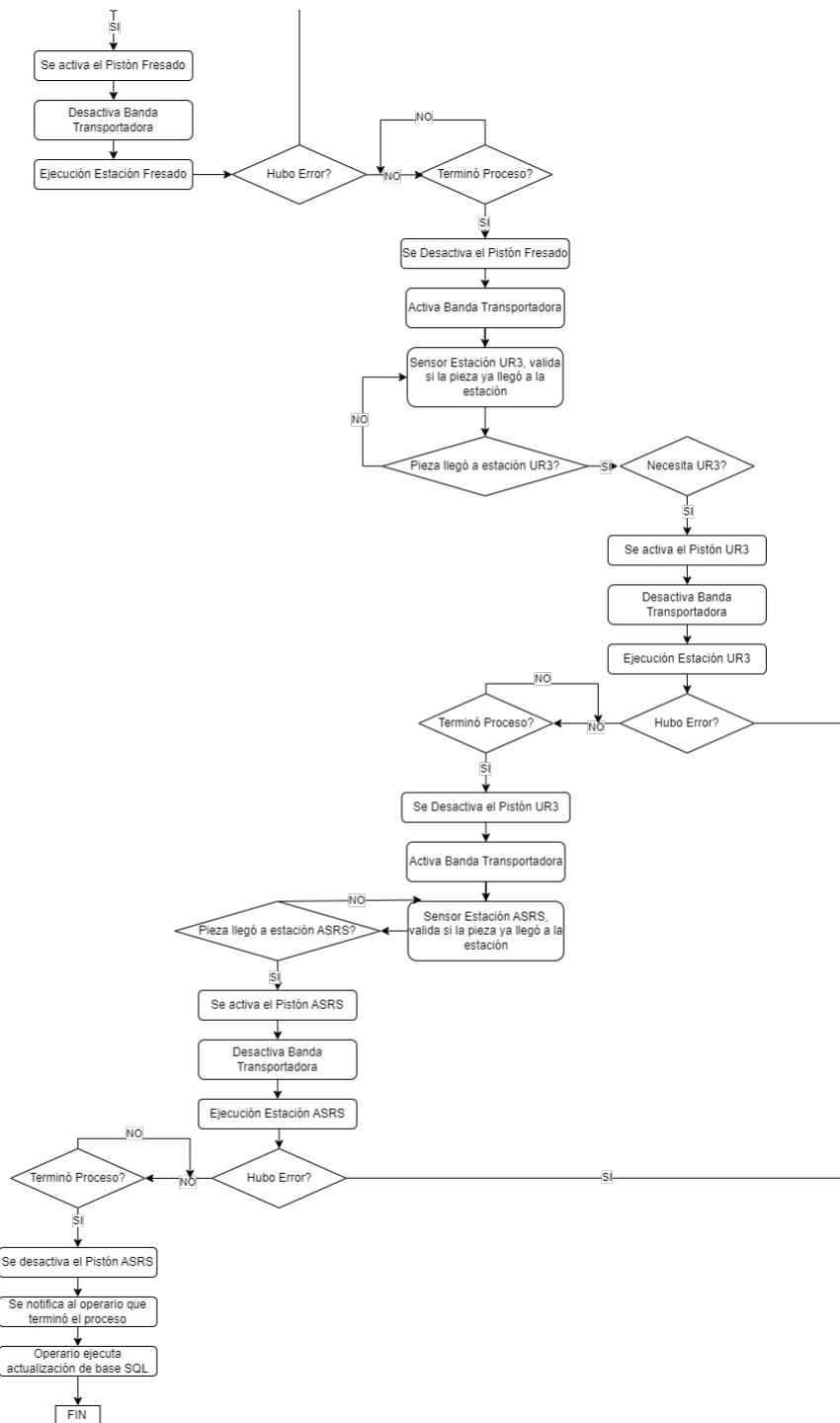


Figura 6.10: Lógica Funcional Celda de manufactura CAP Parte 2

6.2. Especificación de Requisitos

Al conocer el proceso actual del funcionamiento de la celda de manufactura del CAP, se logró establecer unos requerimientos básicos para el diseño del sistema de información:

1. El sistema podrá ser parametrizado para posibles aplicaciones en otras celdas de manufacturas flexibles.
2. El sistema de información deberá contar con una base de datos adaptable para incluir futuros cambios en la topología de la planta.
3. El sistema de información deberá implementar una interfaz de usuario que facilite el control de la celda de manufactura.
 - a) El sistema de información deberá implementar una interfaz de usuario para la creación, modificación y eliminación de órdenes.
 - b) El sistema de información deberá implementar una interfaz de usuario que muestre el detalle de las órdenes creadas.
 - c) El sistema de información deberá de implementar una interfaz de usuario que permita re-establecer la cantidad de materia prima disponible. Es decir, el usuario al acceder a esta opción, volverá a llenar el ASRS con la cantidad disponible por default (15 : Empack, 10: Aluminio).
 - d) El sistema de información deberá de implementar una interfaz de usuario que permita la activación de la celda de manufactura.
 - e) El sistema de información deberá de implementar una interfaz de usuario que cuente con un botón de parada de emergencia.
4. El sistema de información deberá de implementar un software de control encargado de la ejecución automática de la celda de manufactura.
 - a) El sistema de información deberá implementar un software de control que inicie el proceso de manufactura al activar la celda de manufactura flexible del CAP mediante la GUI.
 - b) El sistema de información deberá implementar un software de control que reporte las fallas y el tipo de error presentado en el proceso.
 - c) El sistema de información deberá implementar un software de control que establezca la conexión con la base de datos, para la generación de consultas necesarias para la obtención de datos para el flujo del proceso.
 - d) El sistema de información deberá implementar un software de control que implemente una lógica condicional frente a la disponibilidad de materia prima, agotamiento de producto, mantenimiento de máquinas y notificación de errores.
 - e) El sistema de información debería implementar un software de control que priorice las órdenes a partir de las características definidas.

5. El sistema de información deberá calcular y publicar los indicadores de Eficacia General del Equipo, Tiempos de Operación, Tiempos de Inactividad y Número de productos producidos.

Diseño del Sistema de Información

En la etapa de diseño del sistema de información, se abarcaron cuatro puntos de gran importancia para completar con los requerimientos definidos. Estos puntos son:

- Diseñar una base de datos que permita mantener la lógica y estructura de funcionalidad de la celda de manufactura flexible, brindando a su vez adaptabilidad y versatilidad.
- Diseñar un Dashboard que permita ilustrar los indicadores industriales establecidos y facilitar la toma de decisiones frente al análisis del proceso.
- Diseñar un software de control que permita controlar el funcionamiento físico de la celda, estableciendo una comunicación entre celda y base de datos.
- Diseñar una interfaz gráfica como medio de iteración entre el usuario y el coordinador.

7.1. Base de Datos

7.1.1. Diseño conceptual

Como se mencionó en la introducción del proyecto, el desarrollo de la base de datos se realizará a partir del lenguaje Cypher y de la plataforma de desarrollo Neo4j. Para el primer diseño, se pensó en la representación de máquinas y elementos existentes de la celda, como un nodo en la base de datos. Es decir, la celda cuenta con estaciones, sus estaciones con máquinas, las máquinas fabrican piezas y estas piezas necesitan de material disponible.

Estas relaciones, son las que decidimos optar como las etiquetas que permitieran conectar a los nodos. Por tal razón, el primer diseño de la celda se presenta en la figura 7.1:

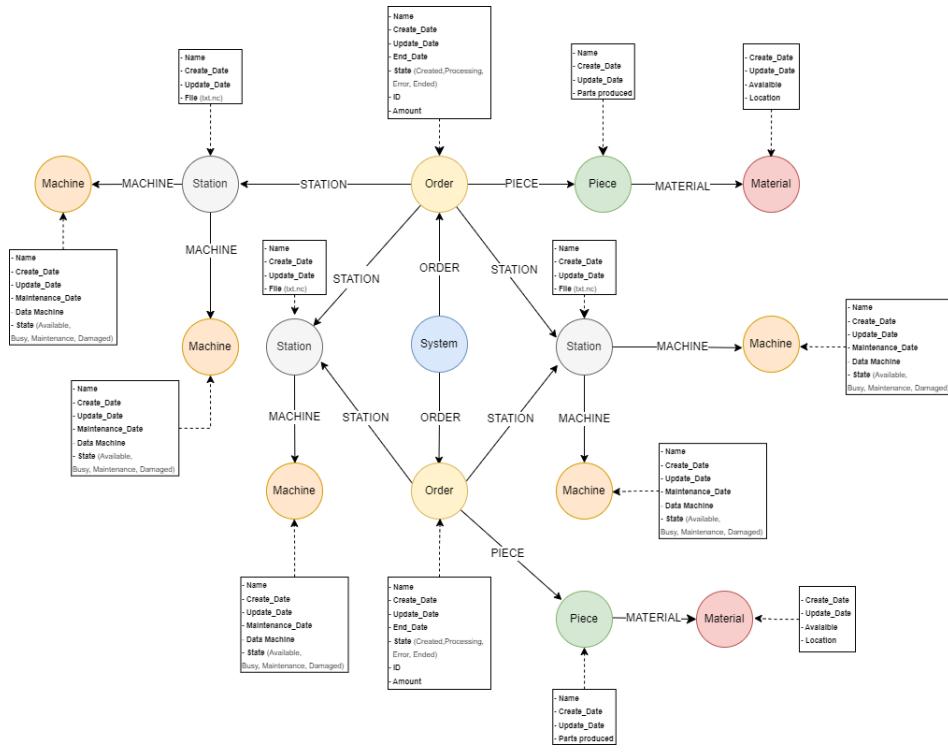


Figura 7.1: Boceto No.1 Base de Datos

Como podemos ver en la figura 7.1 , se definieron los nodos tipo:

- Máquinas (Nodos Naranjas)
- Estaciones (Nodos Grises)
- Orden (Nodos Amarillos)
- Sistema (Nodo Azul)
- Material (Nodos Rojos)
- Piezas (Nodos Verdes)

Las relaciones creadas, tienen el mismo nombre de los nodos. Cabe aclarar que en la figura 7.1, el nombre que se reflejará en cada nodo estará dado por las características de cada Máquina, Estación, Material, entre otros. Por ejemplo: Nodo tipo Maquina tendrá nombre : CNC Torno.

Frente a la visualización de la base de datos, se permitió conseguir una fácil interpretación, en cuanto a la estructura física de la celda y la relación entre sus componentes.

Se determinó que todos los nodos que conformen la base tendrán por default la fecha de creación y de actualización. Esto con el fin de cumplir con etiquetas claves y básicas de la información que brinda una base de datos.

Frente a este primer boceto, se analizó el cómo sería su acople a la celda de manufactura y al dashboard, y si el diseño era suficiente para el funcionamiento actual de la celda de manufactura.

A partir de eso, se generaron los siguientes puntos para tener en cuenta al alimentar el dashboard y poder obtener los indicadores establecidos:

- Producción dada por las órdenes creadas y la cantidad de piezas a realizar.
- Tiempo activo de las máquinas.
- Tiempo presupuestado del uso de las máquinas.
- Fallos y tipo de fallo de la máquina.

Por ende, se optó por añadir esta información a partir de nuevas relaciones entre la máquina implicada y la orden creada, en donde dicha relación guardará esta información en formato JSON o lista para acceder a él y alimentar los indicadores.

Por otro lado, al presentar el primer boceto a nuestros directivos y los expertos del CAP, resaltaron que el diseño realizado, no tiene en cuenta que tanto el material y el tipo de pieza a realizar, afecta en el archivo .NC que la máquina CNC debe de ejecutar. Es decir, como se logra ver en la figura 7.1, el nodo **estación** es aquel que me permite acceder al archivo .NC del proceso. Pero inicialmente, se pensó que era el mismo archivo para la ejecución de la CNC sin depender de la pieza a fabricar y del material.

Debido a esto, se decidió no sólo contar con un sólo archivo .NC por pieza y material disponible, si no que se contará con x archivos donde $x = p * m$, donde p es el número de piezas y m es el número de tipos de materiales.

7.1.2. Depuración del diseño

A partir de la primera corrección por parte de los expertos del CAP y de los directivos, se procedió a generar un segundo diseño de la base de datos, con el fin de que la lógica y estructura lograra abarcar tanto las correcciones realizadas, como el objetivo y funcionalidad de la base.

Dado esto, en la figura 7.2 se muestra el diseño resultante:

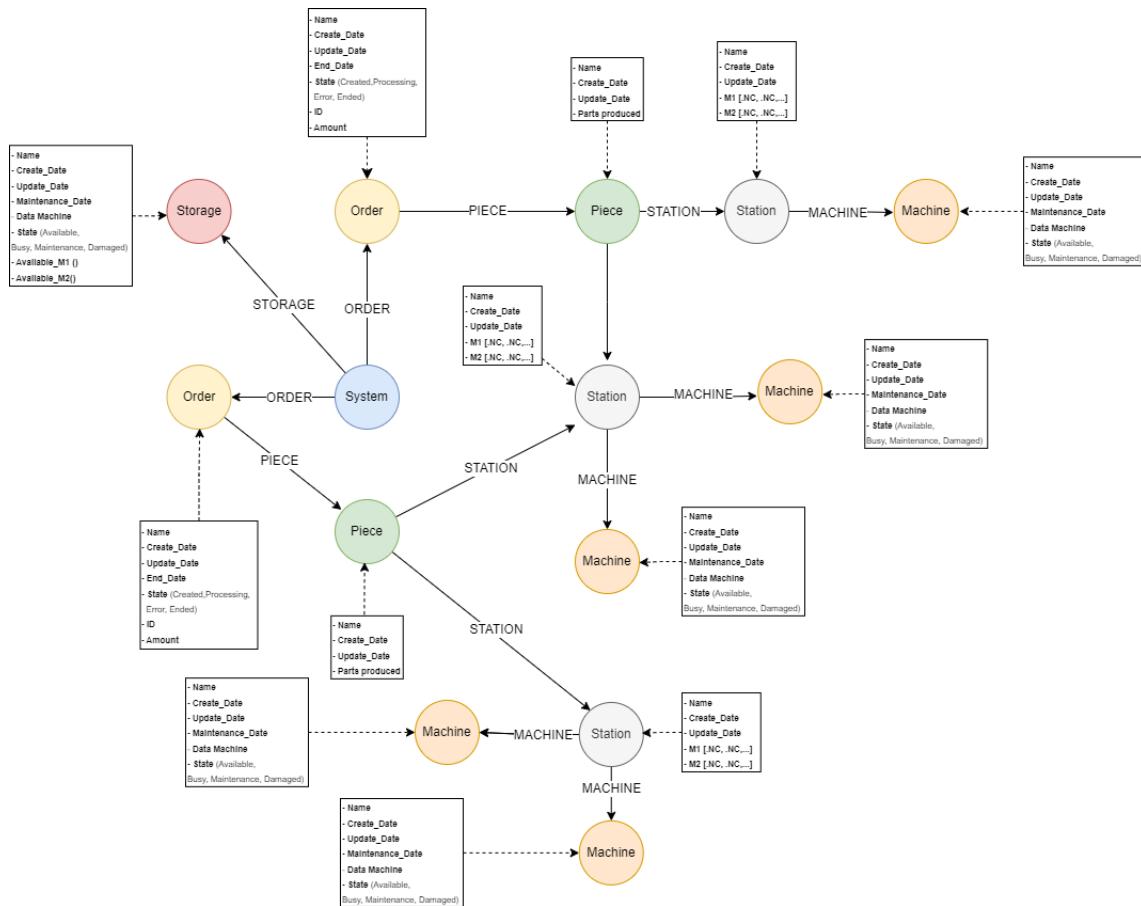


Figura 7.2: Boceto No.2 Base de Datos

Donde los principales cambios realizados se encuentran en la eliminación de los nodos tipo **Material**, dado que este nodo aportaba únicamente la cantidad disponible, por lo que se decidió guardar esa información en el nodo **Storage**.

Por otro lado, en los nodos tipo **Station**, se añadió 2 tipos de propiedades, una referente a los archivos que se deben de ejecutar por el material Aluminio, y el otro, correspondiente al material Empack. Es decir, se cuenta con 2 estaciones (Lathe y Melling) donde se implementa una máquina

CNC para generar el proceso. Además, contamos actualmente con el diseño de 3 piezas, lo que implica que por estación, cada material tendrá 3 archivos diferentes por ejecutar; dando un total de 6 archivos por material.

Al presentar las modificaciones realizadas, se recibieron diferentes correcciones y recomendaciones por parte de los directivos. Entre ellas están:

- No es conveniente establecer la dirección de los archivos .NC como una propiedad guardada en listas. Esto principalmente porque la base de datos perdería la adaptabilidad a otro tipo de celdas de manufactura. Es decir, en este momento fue pensada a las 3 piezas y 2 materiales con los que se cuentan. Por eso, el agregar esos archivos en una propiedad dificulta su adaptabilidad.

El problema está cuando ya no sean 3 piezas si no **p** piezas, o **m** materiales. Esto implica que el nodo **Station** tendrá que añadir **x** propiedades con **p*m** archivos en formato de lista.

- Se recomendó idear un nuevo boceto, donde dejáramos a un lado la arquitectura ya desarrollada, y se tratará de diseñar una completamente nuevo donde se pueda reflejar el funcionamiento del proceso en vez de la composición de la celda. Es decir, cada nodo representará una serie de tareas o pasos que se deben llevar a cabo para ejecutar el proceso.

7.1.3. Boceto Final:

A partir de las recomendaciones y modificaciones dadas, se planteó un nuevo boceto desde cero, tratando de visualizar la base como una serie de pasos o tareas a ejecutar para que el proceso tenga éxito. Esto implica, que a partir de la lógica del funcionamiento de la celda de manufactura 6.9, existen una serie de pasos que se ejecutan para que una pieza P con material M pueda realizarse.

A partir del diagrama lógico, se obtuvieron 2 diseños pre-eliminarios para la base de datos. El primer boceto consiste en:

- Se volvió a considerar el nodo **Estación**, cuyo nodo estará compuesto a su vez por los nodos **Máquinas**. Esto con el fin de volver a representar la estructura de la celda y con la adaptabilidad a cualquier otro tipo de celda, ya que la base de una celda de manufactura se centra en una serie de estaciones compuestas por un equipo de máquinas.
- Los nodos **Pieza** estarán relacionados a una serie de pasos que se deben de ejecutar para conseguir la realización de esta.
- Se creó una nueva relación denominada **NEXT**, indicando el paso siguiente que se debe de ejecutar para la realización de la pieza.

- Se eliminó el nodo **System** dado que no presenta ningún atributo o información relevante para el sistema.

A partir de esto, el tercer boceto se presentó de la siguiente manera:

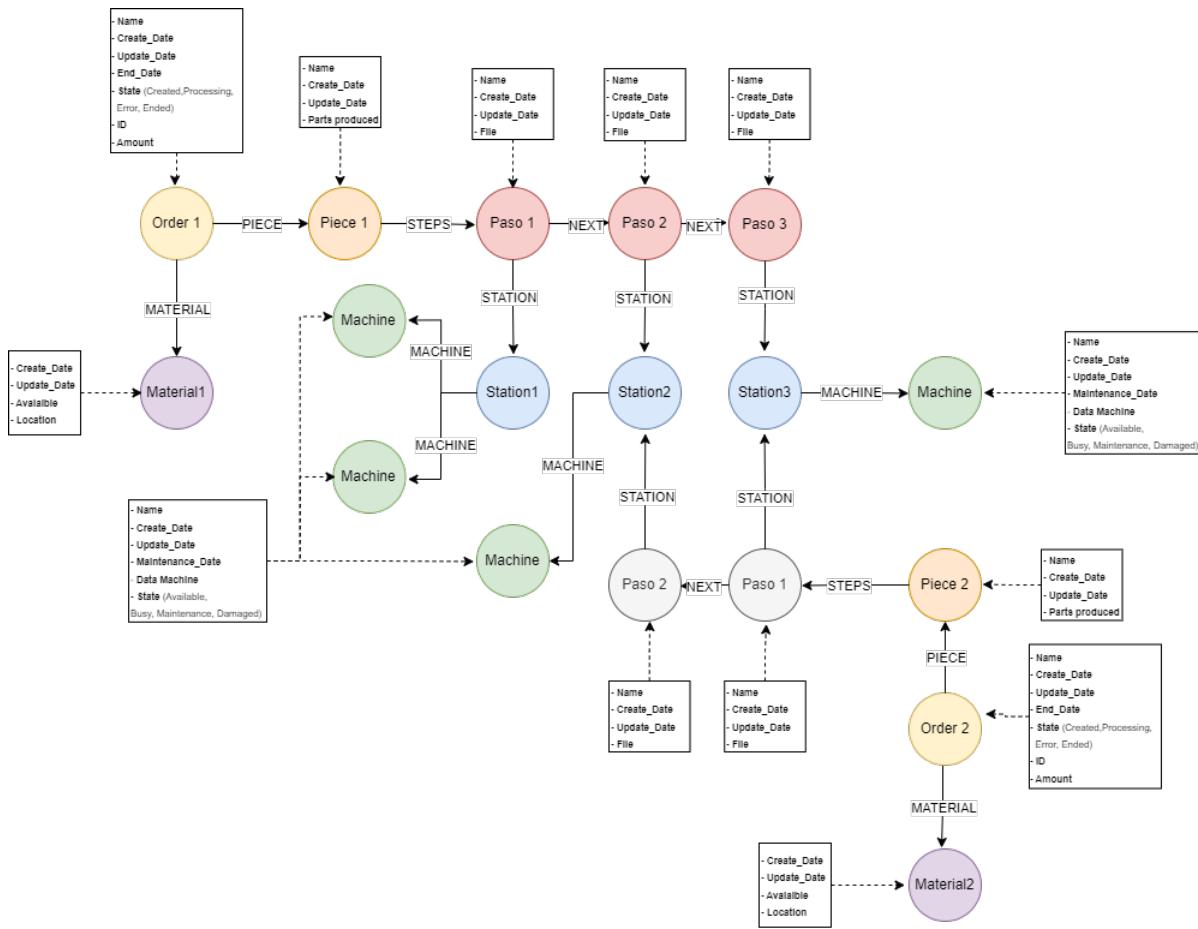


Figura 7.3: Boceto No.3 Base de Datos

Como podemos ver en la figura 7.3 , se definieron los nodos tipo:

- Máquinas (Nodos Verdes)
- Estaciones (Nodos Azules)
- Orden (Nodos Amarillos)
- Pasos tipo A (Nodos Grises)

- Pasos tipo B (Nodos Rojos)
- Material (Nodos Morados)
- Piezas (Nodos Naranjas)

En cuanto a este diseño, se tuvo en cuenta que se puede adaptar a cualquier tipo de celda, más su interpretación se vuelve cada vez más compleja si se agranda el catálogo de piezas y materiales. Esto se debe principalmente que se crearán **n** nodos con diferentes propiedades haciendo referencia a los pasos que se deben de ejecutar, ya que los pasos son diferentes dependiendo de la pieza a producir, y los archivos se modifican dependiendo del material de la pieza.

Por otro lado, el segundo boceto consiste en:

- Analizar los pasos que se deben de ejecutar no como una serie de tareas, si no por estación a la cuál dará uso. Este con el fin de que existen acciones o tareas que consisten en ejecutar la misma estación pero con diferente archivo, por lo que se recurrió al software de control para controlar la activación de las estaciones a partir de las estaciones por las que debe de pasar la pieza, esto implica que los archivos ya no serán un atributo a considerar.
- La etiqueta **STEP** contará con un atributo referente al número de paso y este será utilizado para referenciar la estación y el orden que utiliza la pieza para su producción.
- Se eliminó el nodo **System** dado que no presenta ningún atributo o información relevante para el sistema.

A partir de esto, el boceto No.4 de la base de datos se puede apreciar en la figura 7.4:

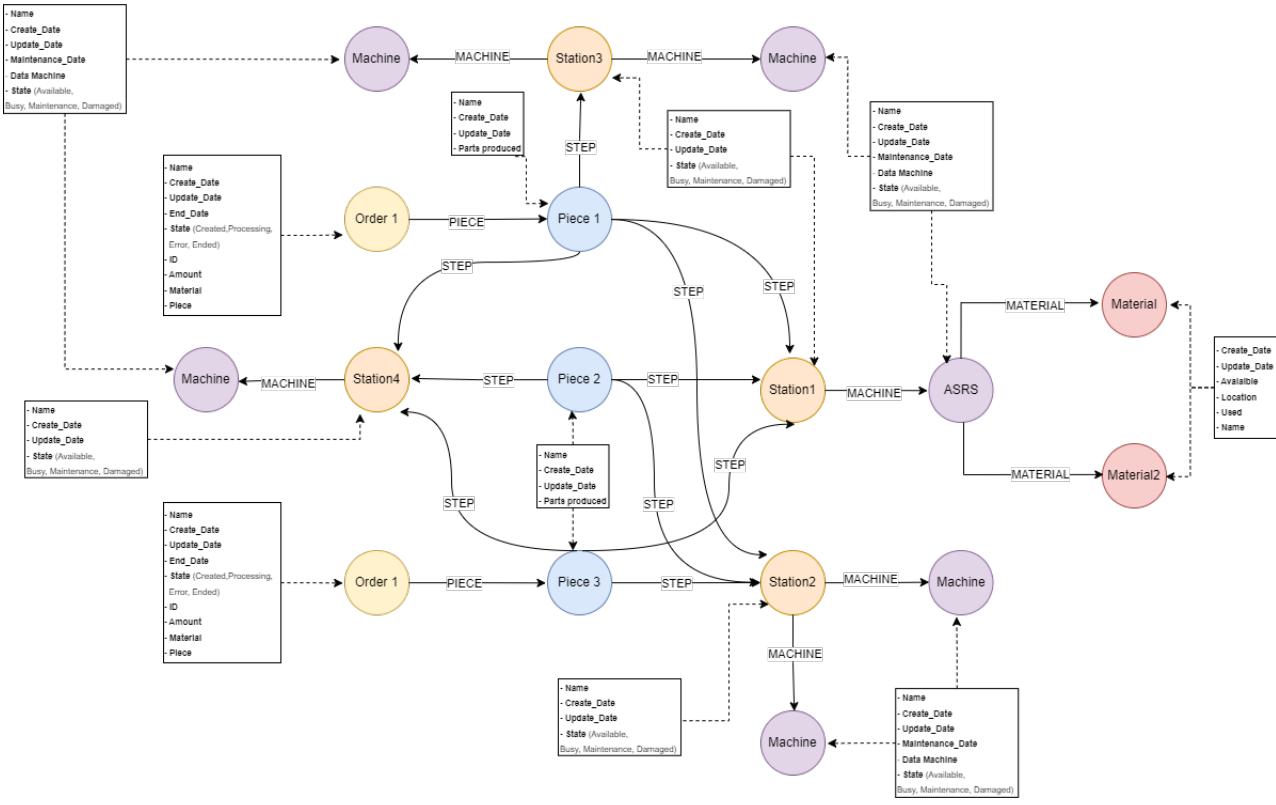


Figura 7.4: Boceto No.4 Base de Datos

Como podemos ver en la figura 7.4 , se definieron los nodos tipo:

- **Máquinas (Nodos Morados):** Estos nodos harán referencia a todas las máquinas presentes en la celda manufactura. En nuestro caso, los robot Mitsubishi, bandas, CNC Torno, CNC Fresado, ASRS y robot UR3.
- **Estaciones (Nodos Naranjas):** El nodo estaciones hará representación de las diferentes etapas de trabajo existentes en la celda de manufactura.
- **Orden (Nodos Amarillos):** Los nodos tipo orden permitirán llevar un registro de órdenes generadas y ejecutadas por parte de la celda de manufactura. Se resalta, que cada Orden sólo puede producir un material en específico.
- **Material (Nodos Rojos):** Estos nodos pretenden guardar la información de los diferentes materiales que se usan para la producción de las piezas. Este nodo brindará información de disponibilidad de piezas.

- **Piezas (Nodos Azules):** El nodo pieza, hace referencia a los diferentes modelos o bocetos de pieza que se pueden realizar. Cabe resaltar, que a partir de este nodo, se definen los pasos que se deben de ejecutar para la realización de la pieza. Del mismo modo, este nodo cuenta con un atributo referente a la cantidad de piezas producidas.

Al presentar los 2 bocetos a los directivos, se determinó que el boceto en el cuál basaremos nuestra base de datos es el Boceto No.4. Esto ya que cumple con los requisitos de ser adaptable a cualquier otro tipo de Celda de Manufactura, y que a su vez, trata de reflejar la lógica funcional de la celda.

Adicionalmente, este boceto permite que al realizar la implementación en otro tipo de celda, o si se presentan modificaciones ya sea agregando/quitando estaciones en la celda, no afectará en la lógica de la creación de una nueva orden, dado que ésta se adaptará a partir de las nuevas características que presente la base de Datos.

7.2. Dashboard

Para el diseño del Dashboard, se tuvo en cuenta que el objetivo es generar una serie de visualizaciones para poder mostrar los siguientes indicadores industriales:

- Tiempos de Operación.
- Tiempos de Inactividad (Downtime).
- Número de productos producidos.

A partir de los indicadores establecidos y de la estructura de la base de datos, se concluyó que la información necesaria para alimentar el Dasboard son:

- Tiempo de ejecución de la pieza por máquina: El tiempo que toma la máquina en realizar una pieza de la orden.
- Tiempo de ejecución total en terminar una orden.
- Total de piezas producidas.
- Tiempo presupuestado de trabajo para la máquina.
- Total de piezas producidas con el detalles de su Status (Error, Creadas, Ejecutando y Finalizadas).
- Información de las órdenes (Material, Cantidad, Status y entre otros.)

Otro punto importante a discutir, es el método o forma en que el dashboard procederá a hacer alimentado, es decir, contará con conexión directa a la base, se dará uso de ETL para la obtención de datos, o se alimentará a partir de un excel que se llenará a partir del coordinador.

Con base en esto, se averiguó sobre las herramientas existentes para generar reportes, en donde se encuentran: Looker Studio, Power BI, Tableau, Graphana y entre otras. Para este caso, se decidió utilizar Looker Studio, debido a su fácil manipulación y facilidad en el diseño estético del reporte.

A partir de esto, se optó por alimentar al Dashboard a través de un Google Sheets, ya que presenta conexión directa con Looker Studio y del mismo modo, este Google Sheets será alimentado por la información que el software de control consulta de la base de datos. Cabe aclarar que Looker Studio también presenta conexión directa a Neo4j, pero para la activación de este Plug-in, se debe tener una subscripción de pago, por lo tanto, se puede retomar esta conexión directa para trabajos futuros.

El diseño generado para el reporte de los indicadores se espera realizar en 5 páginas para ilustrar la información de una forma ordenada y concreta. Lo anterior se visualiza en las siguientes figuras 7.5, 7.6, 7.7, 7.8 y 7.9.

- **Página 1: Reporte de Órdenes**

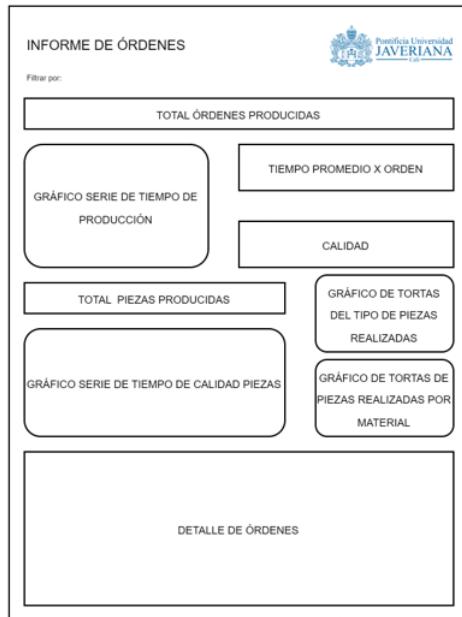


Figura 7.5: Boceto Reporte de Órdenes

- Página 2: Reporte Estación Torno

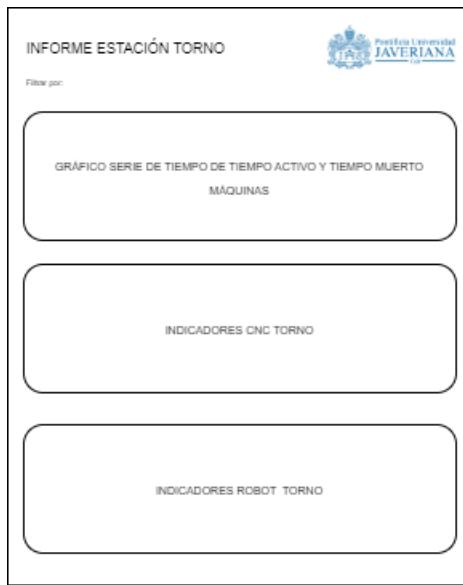


Figura 7.6: Boceto Reporte Estación Torno

- Página 3: Reporte Estación Fresado

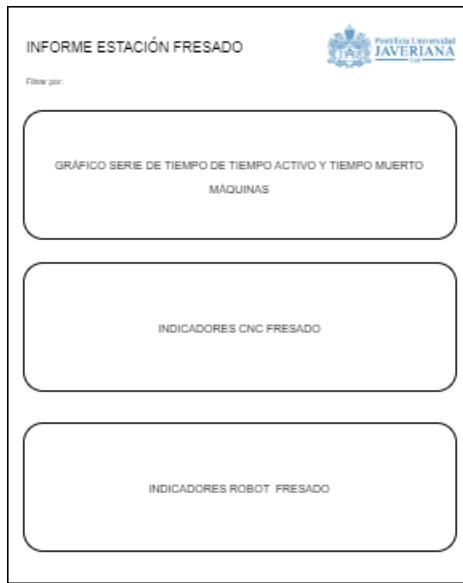


Figura 7.7: Boceto Reporte Estación Fresado

- Página 4: Reporte Estación ASRS



Figura 7.8: Boceto Reporte Estación ASRS

- Página 5: Reporte Estación Inspección



Figura 7.9: Boceto Reporte Estación Inspección

7.3. Interfaz Gráfica (GUI)

El diseño de la Interfaz Gráfica de Usuario (GUI por sus siglas en inglés) surge de la necesidad de proporcionar a los usuarios un punto de interacción con el sistema MES. Por ello, desde el principio se generaron ideas para identificar las principales funcionalidades que se debían llevar a cabo en un proceso de creación y gestión de órdenes dentro de la celda de manufactura y así, plasmarlas en el diseño de la GUI. De lo anterior, se determinaron las siguientes funciones que se visualizan en las figuras 7.10 a la 7.16:

- Crear Orden

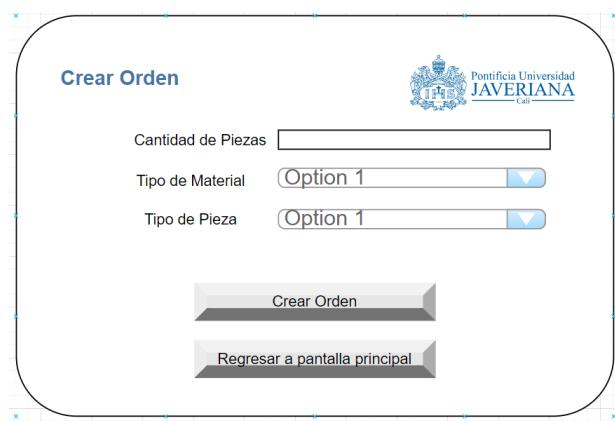


Figura 7.10: Boceto de Crear Orden

- Eliminar Orden



Figura 7.11: Boceto de Eliminar Orden

- Modificar Orden



Figura 7.12: Boceto de Modificar Orden

- Ver Órdenes

Ver Órdenes				
ID de la Orden	Cantidad de Piezas	Tipo de Material	Tipo de Pieza	Fecha de Creación
EP2_2023-08-12_C2	2	Empack	Pieza2	2023-08-12
AP2_2023-08-12_C4	4	Aluminio	Pieza2	2023-08-12
Cerrar				

Figura 7.13: Boceto de Ver Órdenes

- Re- establecer Almacén



Figura 7.14: Boceto de Re-establecer Almacén

- Activar Celda de Manufactura



Figura 7.15: Boceto de Activar Celda de Manufactura

- Botón de Emergencia y Pantalla Inicio

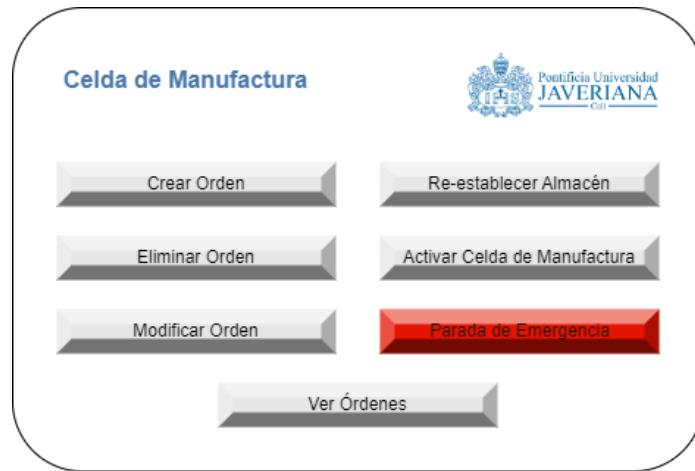


Figura 7.16: Boceto de Botón de Emergencia/ Pantalla Inicio

CAPÍTULO 8

Implementación del Sistema de Información

En la implementación del sistema de información, se diseñó un boceto de cómo sería la comunicación de los diferentes componentes que conforman el sistema (Software de control, Base de datos, Celda de Manufactura y Dashboard), para obtener una mejor interpretación de su funcionalidad. Este boceto se puede apreciar en la figura 8.1.

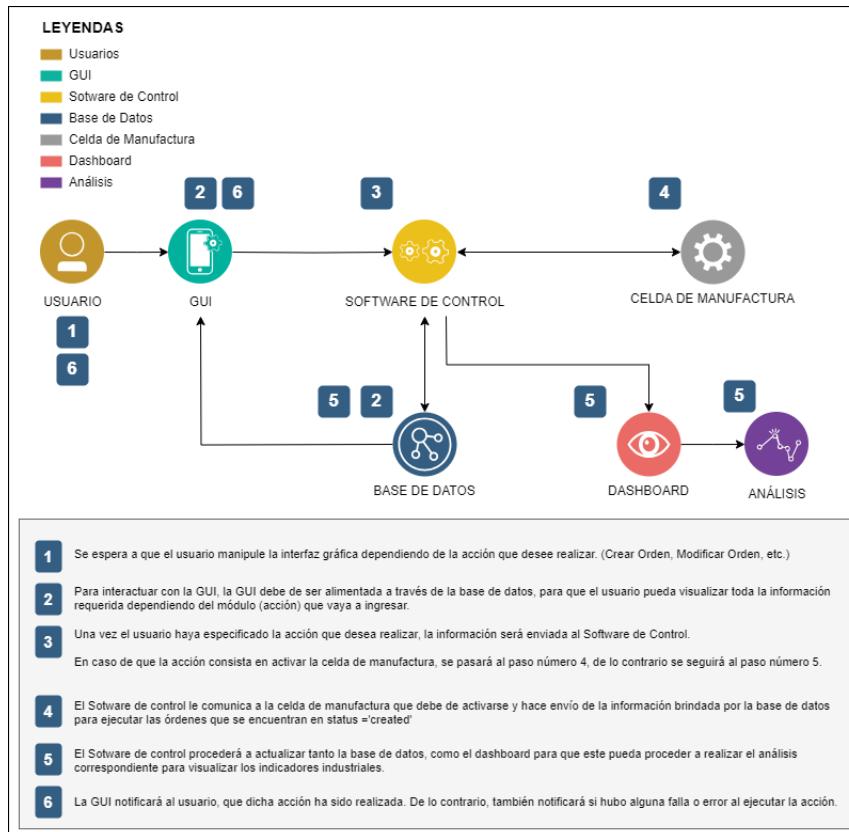


Figura 8.1: Boceto Comunicación Componentes del Sistema de Información

8.1. Base de Datos

Basándonos en el boceto aprobado 7.4, se acudió al gestor de Neo4j para poder generar la base de datos. Para esto, se generó un script en **Python** que permitiera generar las diferentes consultas y modificaciones (información de nodos, creación de nodos, modificación y entre otros) entre el Software de Control y la base de datos en Neo4j.

A partir de esto, se obtuvo como resultado la siguiente base de datos:

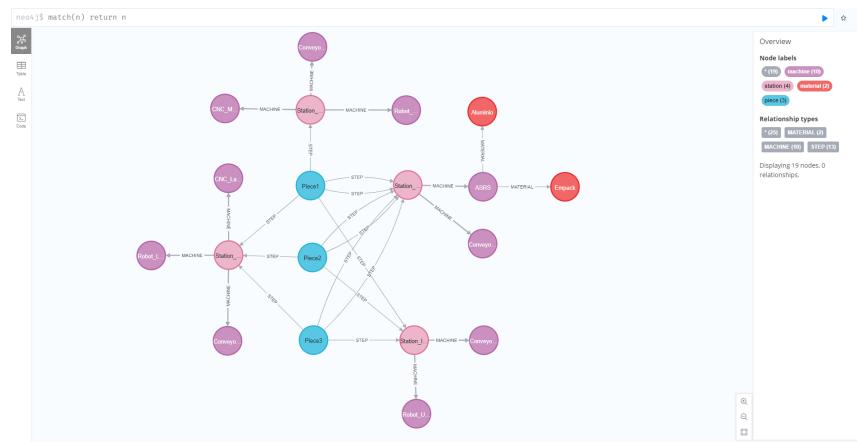


Figura 8.2: Implementación del Boceto No.4 en Neo4j

Como se logra ver en la figura 8.2, se cumple con el diseño aprobado, sin embargo, en esta base de datos aún no se cuenta con los nodos tipo **Ordenes**, debido a que es un ejemplar de la base sin órdenes creadas. Esto con el sentido de que las órdenes se irán añadiendo mediante su creación por medio de la interfaz gráfica (GUI). En el siguiente link: **Diseño Base de Datos**, se encuentra el código implementado para el desarrollo de la base de datos en Neo4j.

A continuación, se detallará la información que brinda cada nodo, y cuántos los constituyen, es decir, cuantas máquinas, estaciones y piezas se cuentan basándonos en la estructura de la Celda de Manufactura del CAP. Como se mencionó anteriormente, cada nodo contará por default la fecha de creación y de actualización:

- **Máquinas** Actualmente existen 10 nodos tipo **Máquinas** referentes a las máquinas que se encuentran disponibles en la Celda de Manufactura del CAP:
 - Máquina CNC Torneadora.
 - Robot Mitsubishi estación Torno.

- Máquina CNC Fresado.
- Robot Mitsubishi estación Fresado.
- Robot UR3 estación Inspección.
- ASRS
- Bandas Transportadoras estaciones (4).

El nodo máquina brindará la información detallada de la marca y modelo de la máquina, su disponibilidad, la estación a la que pertenece y su última fecha de mantenimiento. En la figura 8.3, se puede apreciar los atributos que el nodo contiene:

Node properties >		
machine		
<id>	21	edit
Create_Date	9/23/2023	edit
Data_Machine	Conveyors	edit
Maintenance_	4/11/2025	edit
Date		edit
Name	Conveyor_Lathe	edit
State	Avalaible	edit
Update_Date	9/23/2023	edit

Figura 8.3: Atributos Nodo Máquinas

■ Piezas

A partir de las visitas realizadas al CAP, se determinó que las piezas que actualmente se producen son 3. Cabe aclarar, que no se cuenta con un nombre característico para distinguirlas, por ende, se optó por denominarlas : Pieza 1, Pieza 2 y Pieza 3.

El nodo pieza 8.4, brindará la información de cuantas piezas de ese tipo se han producido:

Node properties >	
piece	
<id>	34
Create_date	9/23/2023
Name	Piece1
Produced	0
Update_Date	9/23/2023

Figura 8.4: Atributos Nodo Piezas

- **Estaciones** Actualmente existen 4 nodos tipo **Estación** referentes a las estaciones que se encuentras disponibles en la Celda de Manufactura del CAP:

- Estación Torno.
- Estación Fresado.
- Estación Montaje/ Inspección.
- Estación ASRS.

El nodo estación brindará el status correspondiente a la estación, es decir, si se encuentra en uso, disponible o entro en falla. A continuación, en la figura 8.5 se muestran los atributos que el nodo brinda:

Node properties >	
station	
<id>	26
Create_Date	9/23/2023
Name	Station_Melling
State	Avalaible
Update_Date	9/23/2023

Figura 8.5: Atributos Nodo Estaciones

- **Material**

La Celda de Manufactura del CAP cuenta con 2 tipos de materiales para la producción : Empack y Aluminio. Por ende, la base actualmente se encuentra constituida por estos dos materiales.

El nodo tipo **Material** brindará información acerca de la disponibilidad, la ubicación en la que se encuentra localizado en el ASRS, y por último, las ubicaciones que ya han sido utilizadas. Este último atributo se añadió para implementar una lógica de control por parte del Software de Control, para indicarle la ubicación al ASRS de donde hay material y no enviarla a una ubicación donde ese material ya fue previamente recogido. Esto permite que la intervención de un operario ya no sea necesaria.

A continuación en la figura 8.6, se muestran los atributos que el nodo material presenta:

Node properties < >		
material		
<id>	33	✎
Available	10	✎
Create_Date	9/23/2023	✎
Location	[[1,1],[1,2],[2,1],[2,2],[3,1],[3,2], [4,1],[4,2],[5,1],[5,2]]	✎
Name	Aluminio	✎
Update_Date	9/23/2023	✎
Used	[]	✎

Figura 8.6: Atributos Nodo Material

8.2. Dashboard: Looker Studio

Basándonos en los prototipos de diseño del reporte, y al acudir a Looker Studio se logró implementar un reporte que se ajustara a los indicadores industriales que se deben de reportar. Cabe resaltar, que este Dashboard será alimentado a través de un script de Python que ejecutará el Software de control alimentando una hoja de Google Sheets, y a su vez este Google Sheets estará conectado al Dashboard.

Para poder ilustrar los indicadores establecidos, se hizo una simulación de los valores que se esperan obtener en la etapa de operar, es decir, alimentamos directamente el Google Sheets para contar con un pre-reporte con la información para el análisis. Se resalta, que aquellos indicadores de tipo tiempo, están establecidos con el formato estándar ISO 8601 que utiliza el sistema de reloj de

24 horas con un formato básico de T[hh][mm][ss]. A continuación, anexamos el resultado obtenido en la elaboración del reporte:

- Página 1: Reporte de Órdenes

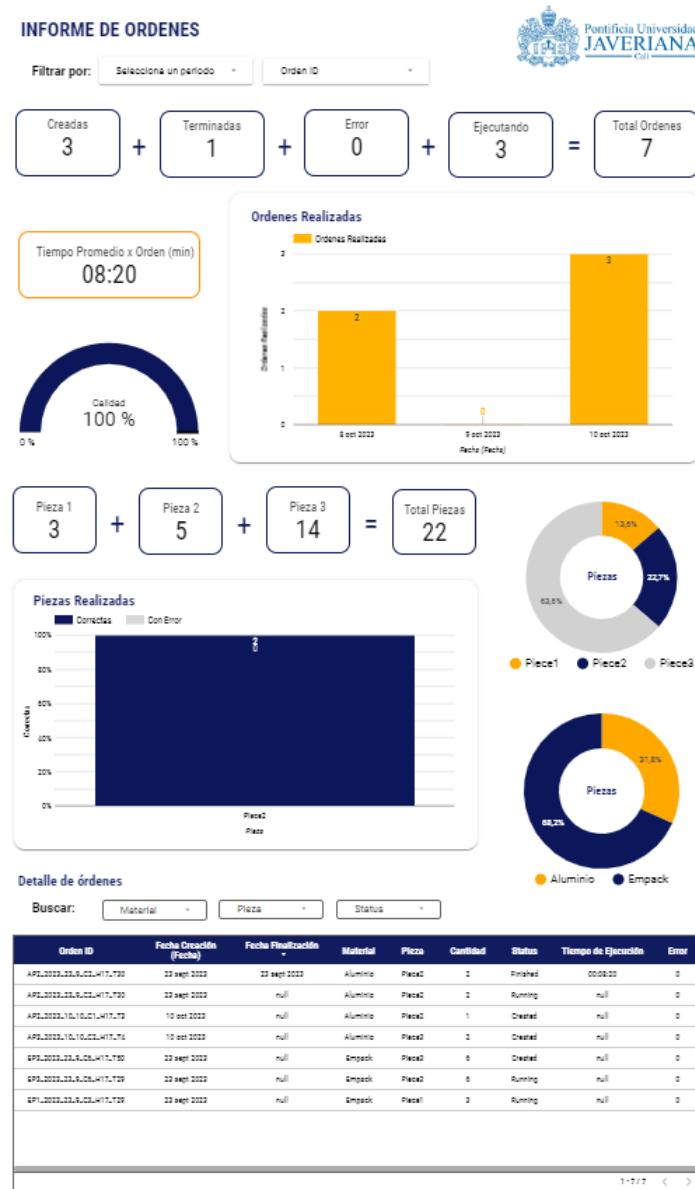


Figura 8.7: Reporte de Órdenes

- Página 2: Reporte Estación Torno



Figura 8.8: Reporte de Estación Torno

- Página 3: Reporte Estación Fresado



Figura 8.9: Reporte de Estación Fresado

- Página 4: Reporte Estación ASRS



Figura 8.10: Reporte de Estación ASRS

- Página 5: Reporte Estación Inspección



Figura 8.11: Reporte de Estación Inspección

- Página 6: Reporte Banda Transportadora

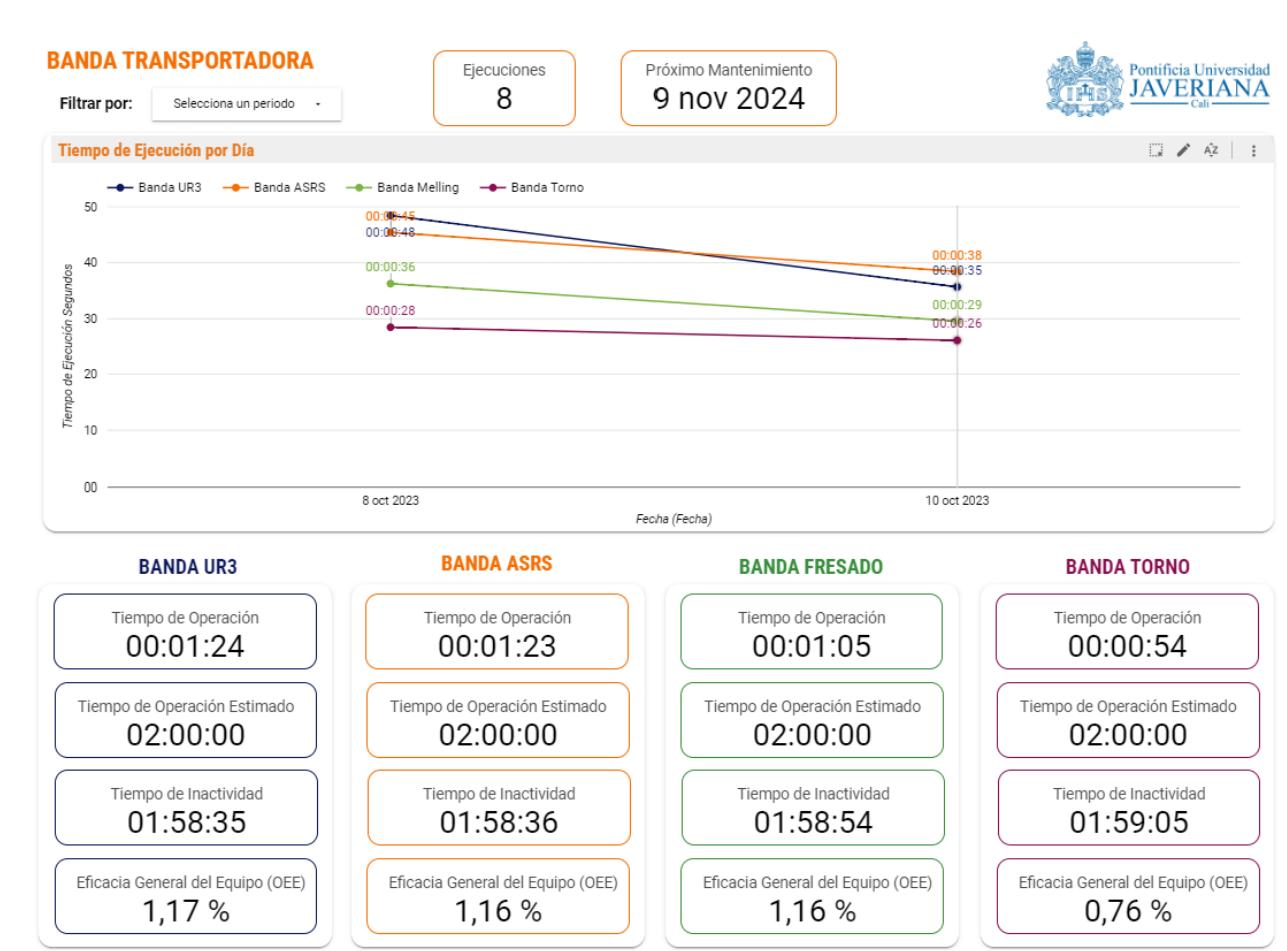


Figura 8.12: Reporte de Banda Transportadora

Para acceder al dashboard generado, acceder al siguiente link: [Dashboard](#)

8.3. Interfaz Gráfica (GUI): Tkinter Python

Luego de haber determinado el diseño de la GUI, se procedió a la implementación mediante el uso de la biblioteca Tkinter y el lenguaje de programación Python. Se programaron funciones para la creación, eliminación, modificación y visualización de las órdenes, así como botones para re-establecer el almacén, activar celda de manufactura y parada de Emergencia.

En la figura 8.13, se puede apreciar la interfaz gráfica desarrollada. Para acceder al código desarrollado de la GUI, ingresar al siguiente link: [Interfaz Gráfica GUI](#).

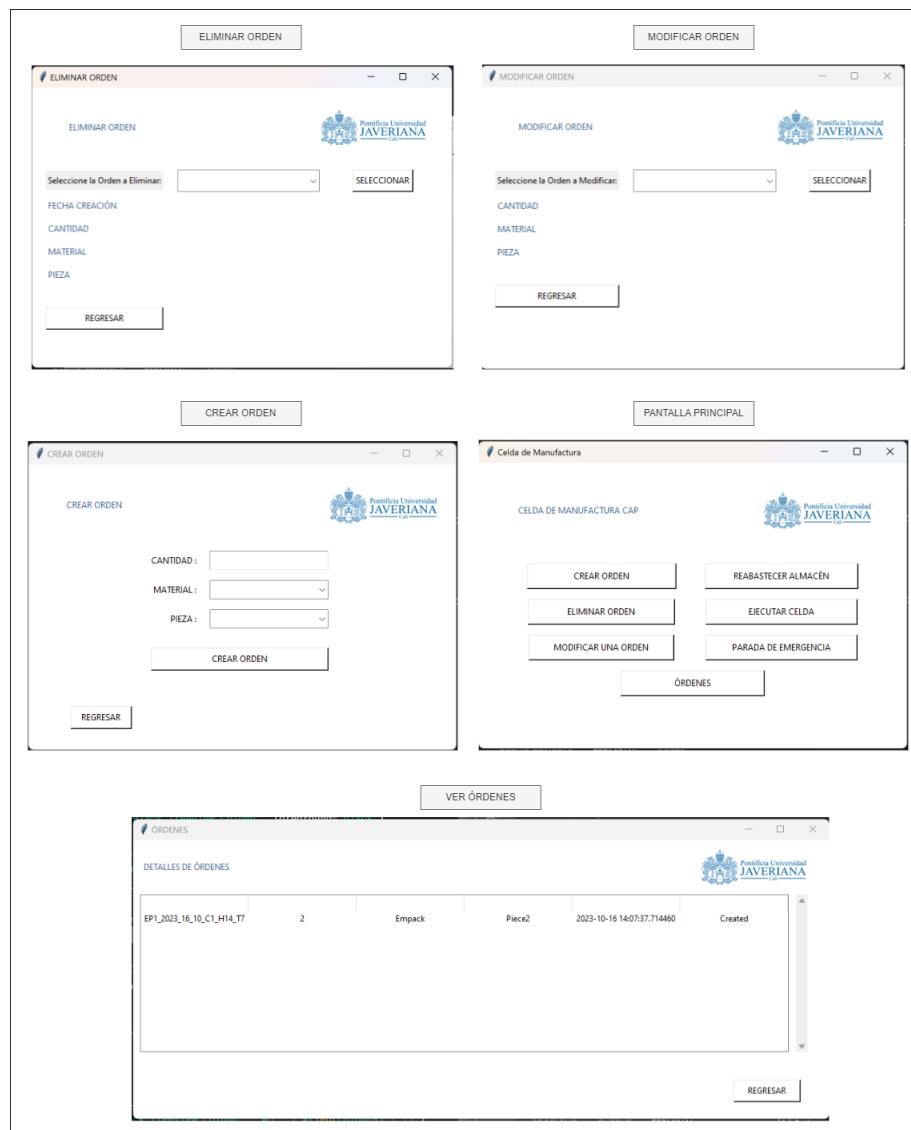


Figura 8.13: Interfaz Gráfica GUI desarrollada en Tkinter Python

8.4. Simulación Celda de Manufactura del CAP: RoboDK

Debido al estado actual de las máquinas y estaciones de la Celda de Manufactura del CAP. No fue posible gestionar una serie de pruebas en un entorno de pedido de manera física. Por esta razón, se decidió informar de este percance a los directos de este proyecto, evaluadores y director de Carrera, para proceder a evaluar la posibilidad de realizar estas pruebas en un entorno simulado. Este percance se detalla a profundidad en la sección de **dificultades** presentadas en el desarrollo del proyecto.

Después de la aprobación de la propuesta, fue necesario acudir a un herramienta de simulación que permitiera realizar una representación simulada de la celda de Manufactura del CAP. Dado esto, se acudió a la herramienta de simulación de procesos industriales RoboDK para desarrollar el diseño. A continuación, se detalla la implementación de esta parte fundamental del proyecto:

8.4.1. Configuración de RoboDK

Para llevar a cabo la simulación de la celda de manufactura, se utilizó la plataforma RoboDK. Esta herramienta se seleccionó debido a su capacidad para simular una amplia variedad de robots industriales y máquinas. La configuración de RoboDK incluyó los siguientes pasos:

- **Selección de los componentes Industriales:** Con el fin de replicar de manera fiel la celda de manufactura del CAP y garantizar que la simulación fuera lo más realista posible, se eligieron los modelos de robots industriales y máquinas. Los principales componentes de la celda de manufactura en RoboDK incluyeron:
 - **Estación de Montaje:** Es utilizada para las tareas de inspección del acabado final de las piezas realizadas en cada orden a través de un brazo robótico colaborativo UR3.
 - **Mitsubishi RV-2FR:** Estos brazos robóticos se integraron en las secciones de torneado y fresado como robots colaborativos para tomar y colocar las piezas dentro de las máquinas de cada estación y sobre la banda cuando han finalizado de procesar cada pieza.
 - **Estaciones de Torneado y Fresado:** Se configuraron las máquinas de torno y fresado con las estaciones de Mazak Lathe y Mazak Melling para replicar las operaciones de moldeado y mecanizado en la celda de manufactura del CAP.
 - **Almacén ASRS:** Se modeló y configuró un almacén automatizado para gestionar el flujo de materiales. Esto incluyó la programación de la transferencia de piezas entre el almacén y las estaciones de trabajo. Por otro lado, no fue posible encontrar un robot cartesiano o similar al real, por lo que se acudió a representar las ubicaciones mediante mesas y robot colaborativo UR3.
- **Configuración de la banda transportadora:** Dentro de la celda de manufactura, se implementó una banda transportadora para el movimiento eficiente de materiales entre las diferentes estaciones de trabajo. La configuración de la banda transportadora incluyó:

- **Diseño de la Ruta de la Banda Transportadora:** Se diseñó una ruta de transporte que conectaba todas las estaciones relevantes, permitiendo un flujo continuo de materiales y piezas de trabajo.
- **Control de la Banda Transportadora:** Se desarrolló un programa para controlar la velocidad y dirección del movimiento de la banda para transportar los materiales entre las estaciones y entregarlos en el momento adecuado para su procesado.
- **Sensores y Detección de Obstáculos:** Se implementaron sensores infrarrojos a lo largo de la banda transportadora para detectar piezas, evitar colisiones y garantizar un proceso de producción seguro y eficiente.

La configuración final del entorno de simulación de la Celda de Manufactura se puede ver en la siguiente imagen 10.5:

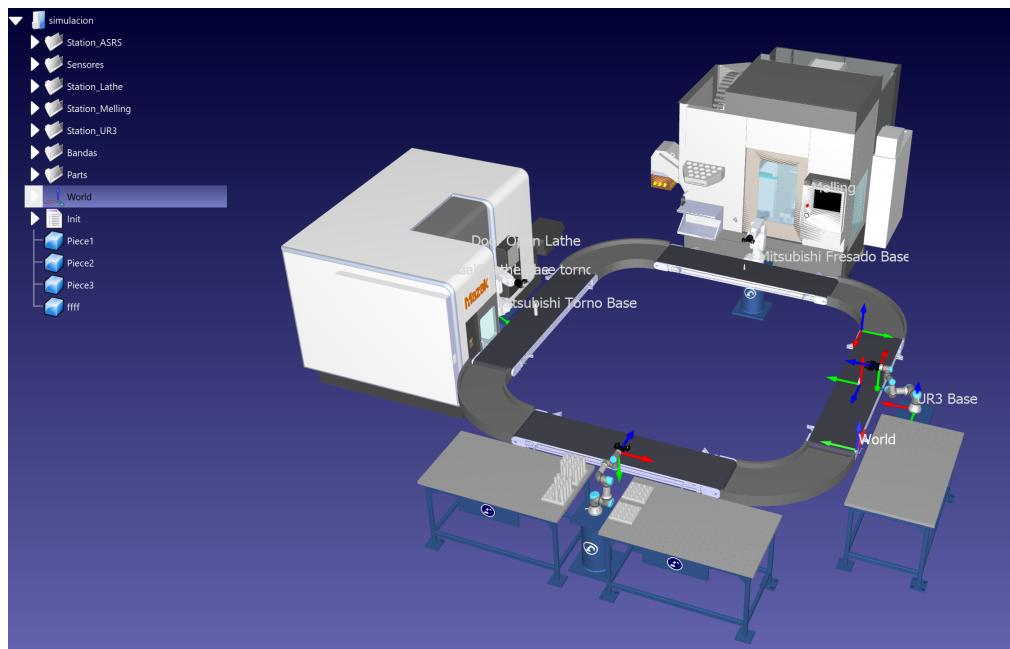


Figura 8.14: Celda de Manufactura en RoboDK

Para acceder a los códigos implementados en la simulación de la Celda de Manufactura, ingresar al siguiente link: [Simulación RoboDK](#).

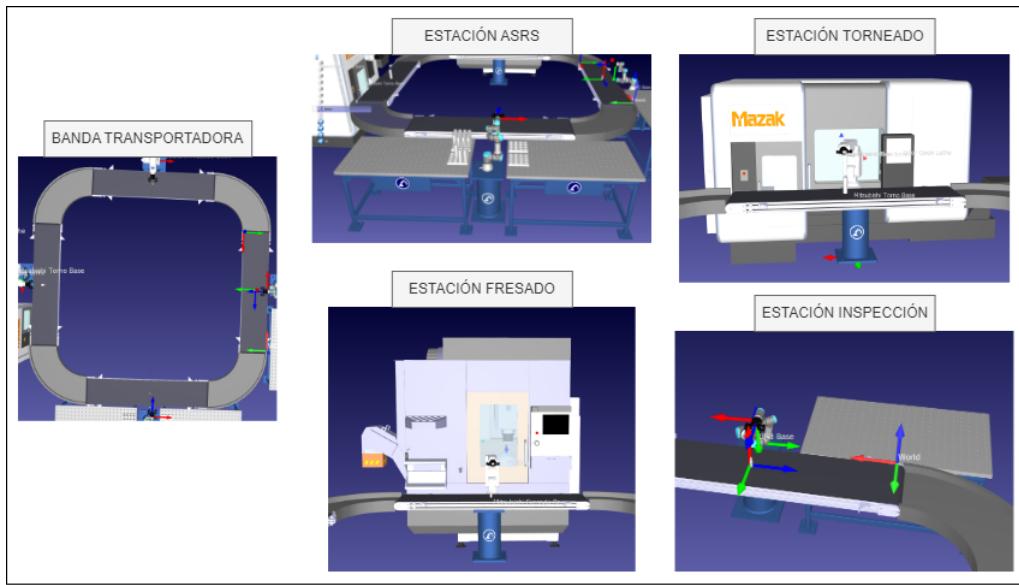


Figura 8.15: Estaciones en RoboDK

8.5. Software de Control

Para la implementación del Software de Control, se optó por desarrollar toda la lógica funcional de la Celda de Manufactura del CAP en un conjunto de **scripts** que permitieran una correcta ejecución.

Esta lógica funcional permitió:

- **Independencia del Operario:** El operario sólo será requerido para la manipulación de la Interfaz Gráfica y para la intervención en las estaciones y/o maquinarias, cuando estas estén en fallo.
- **Control de la Celda de Manufactura:** El software de control tiene la capacidad de activar y desactivar las máquinas a través de comunicación OPC UA, lo que permite a su vez, poder ejecutar en paralelo las estaciones.
- **Ahorro de consumo energético:** El Software de control fue desarrollado de tal manera que activara la banda por secciones y no por completo, lo que reduce consumo innecesario de energía y a la vez, desgaste de la banda.

Dado esto, la lógica funcional a partir del desarrollo en el código de programación de Python se aprecia en las figuras 8.16 y 8.17:

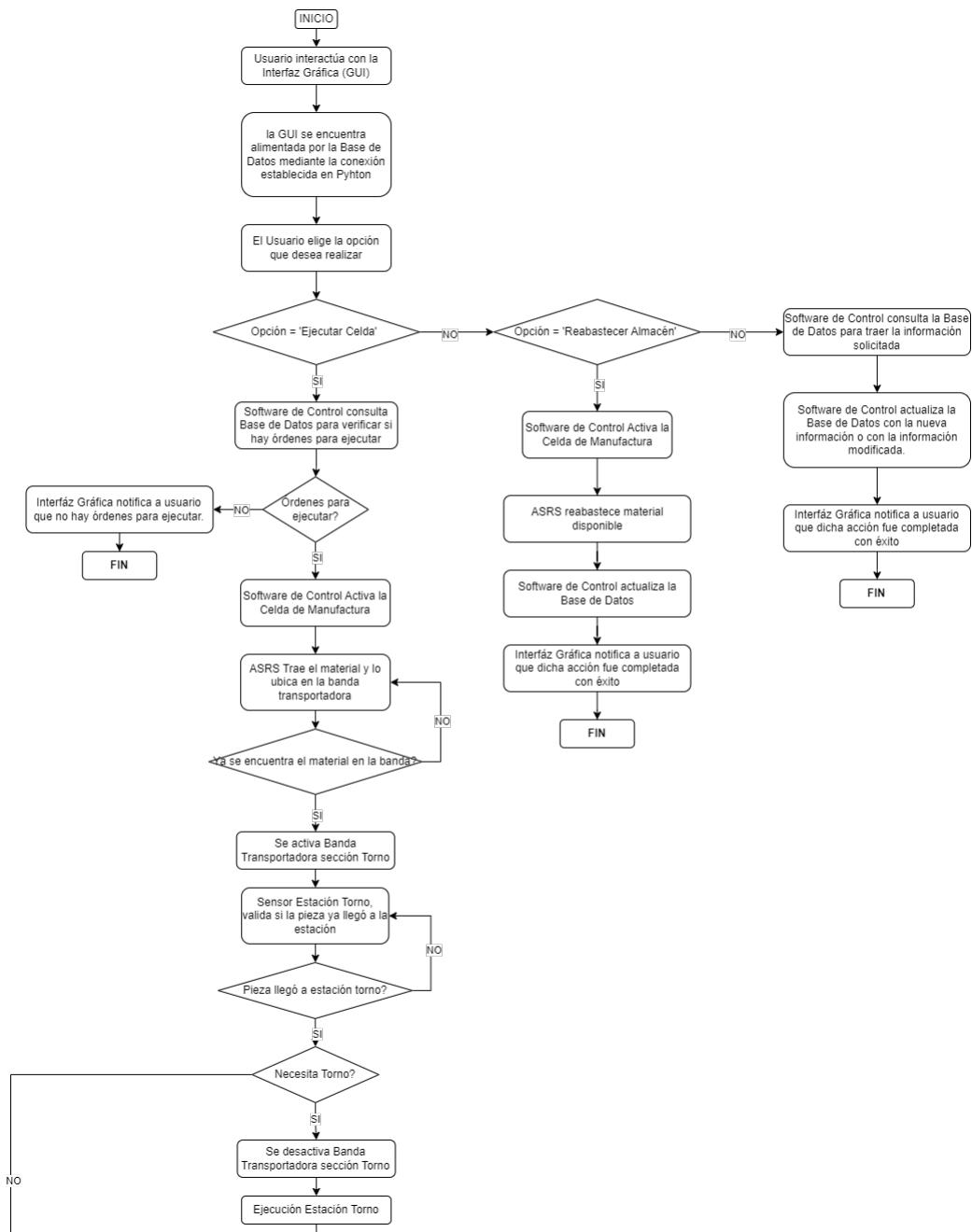


Figura 8.16: Lógica funcional Software de Control Parte 1



Figura 8.17: Lógica funcional Software de Control Parte 2

El Software de Control, está constituido por una serie de scripts que permiten el control de la Celda de Manufactura, modificación y/o comunicación tanto con la base de datos, como con el Dashboard. Para acceder a los diferentes scripts generados, ingresar a los siguientes links:

- **Software de Control**
- **Script de consultas a Base de Datos**
- **Script de comunicación a Dashboard**

8.6. Validación de Conexión y Comunicación (Software de Control - RoboDK - Dashboard - Base de Datos y GUI)

En la validación de la etapa de implementación, se optó por realizar una serie de pruebas básicas, con el fin de validar la comunicación y conexión entre los diferentes módulos diseñados (RoboDk-Software de Control - Base de Datos - Interfaz Gráfica y Dashboard).

8.6.1. Pruebas Botón Crear Orden

Para dar inicio a la prueba de Crear Orden, se definieron las siguientes características que tendrá la orden:

- **Material:** Empack
- **Cantidad:** 3
- **Pieza:** Pieza 2

A continuación, se describirán los pasos que se deben de ejecutar para crear una orden:

- **Paso 1:** Damos clic al botón **Crear Orden**, este nos dirigirá a una nueva ventana para crear la orden a partir de las características deseadas.
- **Paso 2:** Ingresamos la cantidad de piezas deseadas a producir.
- **Paso 3:** Se selecciona el tipo de material que se desea. Se resalta que la información que se despliega es directamente alimentada por la base de datos, por lo que si se añade un nuevo material o se modifica, automáticamente aparecerá y actualizará en el despliegue.
- **Paso 4:** Del mismo modo, se procede a definir el tipo de pieza que la orden producirá.
- **Paso 5:** Damos clic en **Crear Orden** y automáticamente despliega una nueva ventana con la información de la orden creada. Cabe resaltar que el ID de la Orden estará dado por las características ingresadas y la hora de creación de la orden.

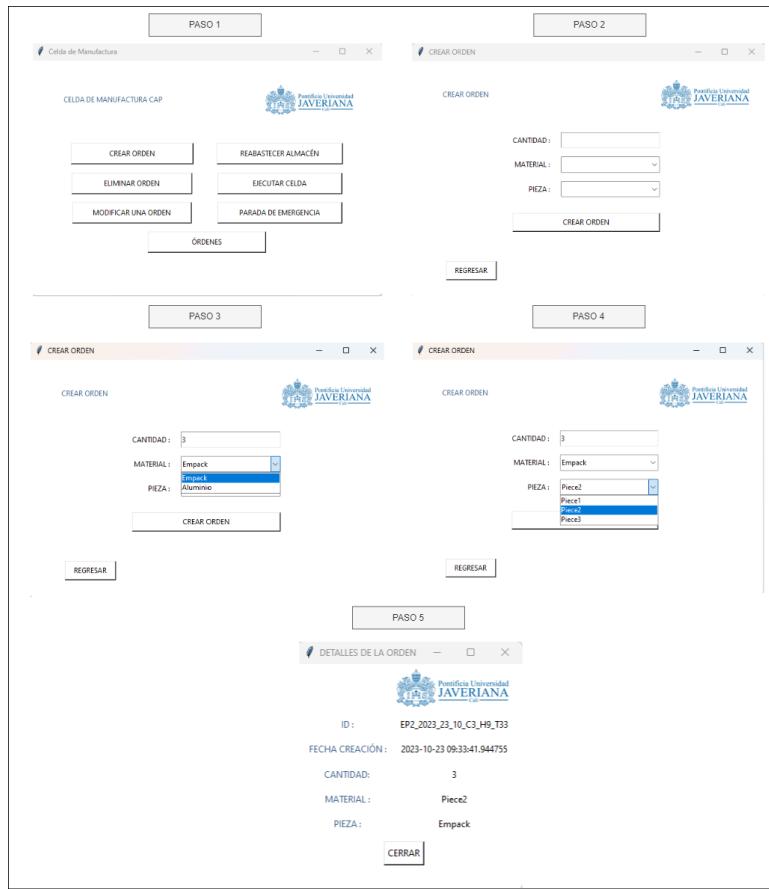


Figura 8.18: Secuencia de pasos para Crear una Orden



Figura 8.19: Descripción de ID Orden

En la figura 8.18, se muestra visualmente los pasos que se deben de ejecutar para crear una orden.

Al crear la orden, se generó el siguiente ID detallando la siguiente información en la figura 8.19: Por otro lado, para validar la conexión con la base de datos y el Dashboard, recurrimos a actualizar la pantalla, obteniendo la siguiente figura 8.20:

En la figura 8.20, vemos como el nodo color **café** se añade a la base de datos, este nodo hace referencia a todos los nodos tipo **orden**. Por ende, los atributos que brinda el nodo tipo orden se

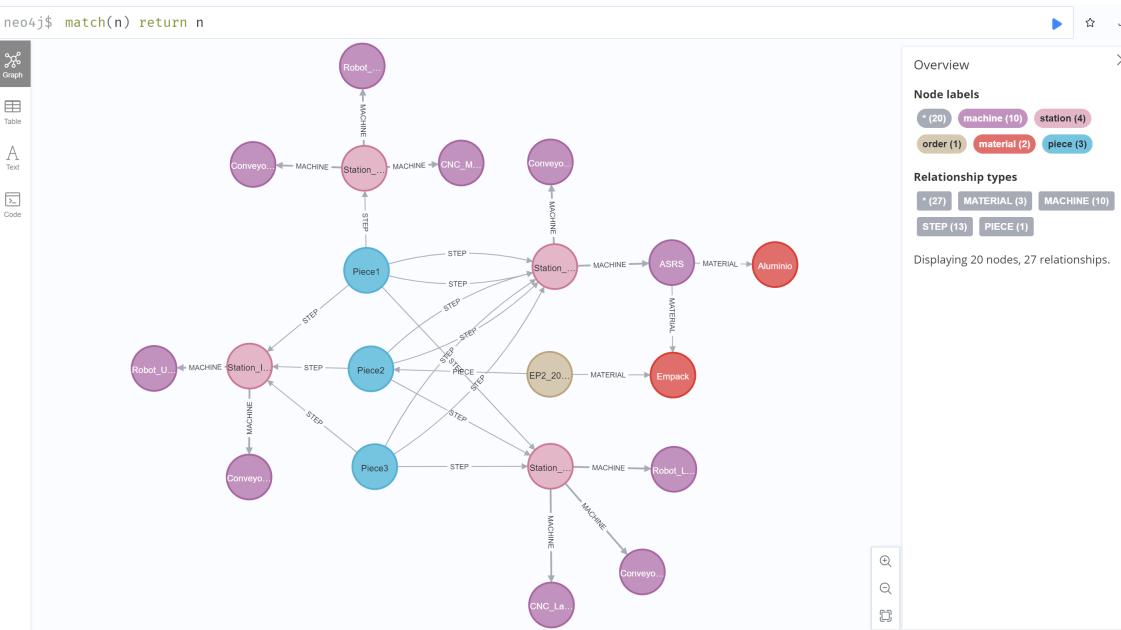


Figura 8.20: Actualización Base de Datos

aprecian en la figura 8.21:

Node properties	
order	
<id>	36
Amount	3
Amount_Error	
Create_Date	2023-10-23 09:33:41.944755
End_Date	
ID_Order	EP2_2023_23_10_C3_H9_T33
Locations	[[], [1, 3], [1, 4], [1, 5]]
Material	Empack
Name	EP2_2023_23_10_C3_H9_T33
Piece	Piece2
State	Created
Type_Error	
Update_Date	2023-10-23 09:33:41.944755

Figura 8.21: Atributos Nodo Tipo Orden

Por otro lado, al crear una **nueva orden**, el material debe actualizar debido a que este dispone de su disponibilidad a las órdenes ya creadas, es decir, la cantidad de piezas que contiene esa orden, afecta a la cantidad restante disponible. Por ende, al crear una nueva orden con **3** piezas de material **Empack**, el material se debe de reducir 12, dado que son 15 las disponibles en el almacén.

Como podemos ver en la figura 8.22, no sólo actualiza la disponibilidad, si no también las ubi-

Node properties >	
material	
<id>	29
Available	12
Create_Date	9/23/2023
Location	[2, 3], [2, 4], [2, 5], [3, 3], [3, 4], [3, 5], [4, 3], [4, 4], [4, 5], [5, 3], [5, 4], [5, 5]
Name	Empack
Update_Date	2023-10-23 09:33:41.131580
Used	[[1, 3], [1, 4], [1, 5]]

Figura 8.22: Actualización Nodo Material

caciones utilizadas, ya que informó al sistema de información que para generar dicha orden, se debe de acudir a esa ubicación.

En adición, al refrescar el Dashboard se obtuvo la figura 8.23. Cabe resaltar que el indicador que aparece vacío, se debe a que este ilustra las ejecuciones de las órdenes, pero como actualmente no se ha ejecutado la celda, este no cuenta con información a mostrar.

Para acceder al vídeo de la prueba **Crear Orden**, ir al siguiente link: [Prueba Crear Orden](#)

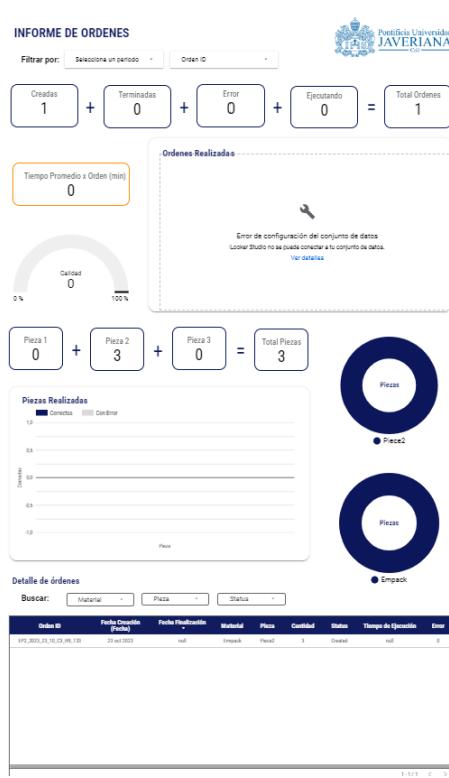


Figura 8.23: Actualización Dashboard

8.6.2. Pruebas Botón Modificar Orden

Para ejecutar la prueba de Modificar Orden, se seguirá con el caso inicial de la orden con ID **EP2_2023_23_10_C3_H9_T33**. En este caso, procederemos a modificarle la pieza y la cantidad a realizar. Dado esto, los pasos a seguir son los siguientes:

- **Paso 1:** Damos clic al botón **Modificar Orden**, este nos dirigirá a una nueva ventana para modificar la orden.
- **Paso 2:** Se abre la nueva ventana de Modificar Orden.
- **Paso 3:** Se selecciona la orden que se desea modificar. En este caso, se resalta que sólo aparecerán órdenes que aún no se han ejecutado debido a que apenas se ejecuta, no es posible modificarla.
- **Paso 4:** Al seleccionar la orden a modificar, la GUI trae toda las características correspondientes a esa orden y el campo para modificarlas.

- **Paso 5:** Se debe de ingresar la información que se desea modificar, en caso de que un campo no se deseé modificar, se debe de ingresar el mismo valor anterior.
- **Paso 6:** Dar clic a **Modificar Orden**, aparecerá un mensaje de advertencia confirmando si está seguro de la modificación a realizar.
- **Paso 7:** Al aceptar la modificación, aparecerá una nueva ventana con un mensaje detallando que la modificación ha sido generada con éxito.

En la figura 8.24, se muestra visualmente los pasos que se deben de ejecutar para modificar una orden.

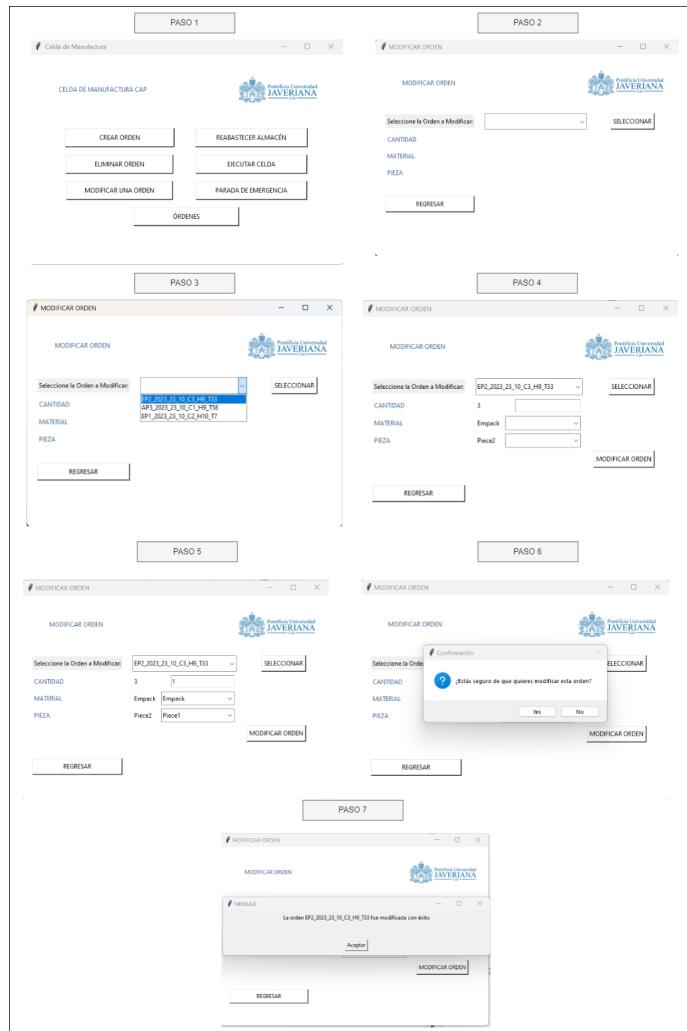


Figura 8.24: Secuencia de pasos para Modificar una Orden

Al validar la conexión con la base de datos y el Dashboard, recurrimos a actualizar la pantalla, obteniendo la siguiente figura 8.25:

Node properties	
material	
<id>	29
Available	10
Create_Date	9/23/2023
Location	[[2, 5], [3, 3], [3, 4], [3, 5], [4, 3], [4, 4], [4, 5], [5, 3], [5, 4], [5, 5]]
Name	Empack
Update_Date	2023-10-23 10:07:55.574376
Used	[[1, 3], [1, 4], [1, 5], [2, 3], [2, 4]]

Node properties	
material	
<id>	29
Available	12
Create_Date	9/23/2023
Location	[[1, 4], [1, 5], [2, 5], [3, 3], [3, 4], [3, 5], [4, 3], [4, 4], [4, 5], [5, 3], [5, 4], [5, 5]]
Name	Empack
Update_Date	2023-10-23 11:51:43.141360
Used	[[1, 3], [2, 3], [2, 4]]

Figura 8.25: Actualización Material Empack

En la figura 8.25 vemos que anteriormente la orden contaba con 3 piezas a producir, por ende, al modificarla a sólo una, automáticamente se encuentran disponibles 2 más. Del mismo modo, se vuelve a dejar disponible las ubicaciones donde dicho material se encuentra. Esta actualización se aprecia en la figura 8.26.

Node properties	
order	
<id>	36
Amount	3
Amount_Error	
Create_Date	2023-10-23 09:33:41.944755
End_Date	
ID_Order	EP2_2023_23_10_C3_H9_T33
Locations	[[1, 3], [1, 4], [1, 5]]
Material	Empack
Name	EP2_2023_23_10_C3_H9_T33
Piece	Pieza2
State	Created
Type_Error	
Update_Date	2023-10-23 09:33:41.944755

Node properties	
order	
<id>	36
Amount	1
Amount_Error	
Create_Date	2023-10-23 09:33:41.944755
End_Date	
ID_Order	EP2_2023_23_10_C3_H9_T33
Locations	[[1, 3]]
Material	Empack
Name	EP2_2023_23_10_C3_H9_T33
Piece	Pieza1
State	Created
Type_Error	
Update_Date	2023-10-23 11:51:43.294767

Figura 8.26: Actualización Orden EP2_2023_23_10_C3_H9_T33

En cuanto a la actualización de la orden **EP2_2023_23_10_C3_H9_T33**, su cantidad pasó a hacer 1 y del mismo modo ya no se espera producir la **Pieza 2**, si no la **Pieza 1**, lo que implica que en la base de datos la relación **PIECE** debe apuntar a **Pieza 1**, 8.27:

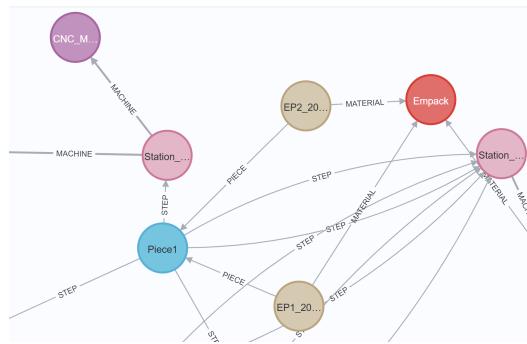


Figura 8.27: Actualización relación PIECE

Por otro lado, al refrescar la información del Dashboard, se visualiza cómo la información se modifica automáticamente. La modificación se puede ver en la figura 8.28.

Para acceder al vídeo de la prueba **Modificar Orden**, ir al siguiente link: [Prueba Modificar Orden](#)

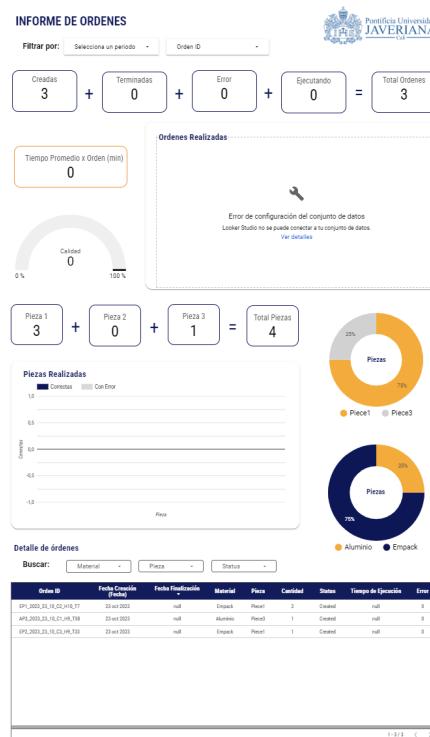


Figura 8.28: Actualización Dashboard

8.6.3. Pruebas Botón Eliminar Orden

Para ejecutar la prueba de Eliminar Orden, se seguirá con el caso de la orden con el ID **EP2_2023_23_10_C3_H9_T33**. Dado esto, los pasos a seguir son los siguientes:

- **Paso 1:** Damos clic al botón **Eliminar**, este nos dirigirá a una nueva ventana para eliminar la orden.
- **Paso 2:** Se abre la nueva ventana de Eliminar Orden.
- **Paso 3:** Se selecciona la orden que se desea eliminar. En este caso, se resalta que sólo aparecerán órdenes que aún no se han ejecutado debido a que apenas se ejecuta, no es posible eliminarla.
- **Paso 4:** Al seleccionar la orden a eliminar, la GUI trae toda las características correspondientes a esa orden.
- **Paso 5:** Dar clic a **Eliminar Orden**, aparecerá un mensaje de advertencia confirmando si está seguro de proceder con la eliminación.
- **Paso 6:** Al aceptar la eliminación, aparecerá una nueva ventana con un mensaje detallando que la eliminación ha sido generada con éxito.

En la figura 8.33, se muestra visualmente los pasos que se deben de ejecutar para eliminar una orden.

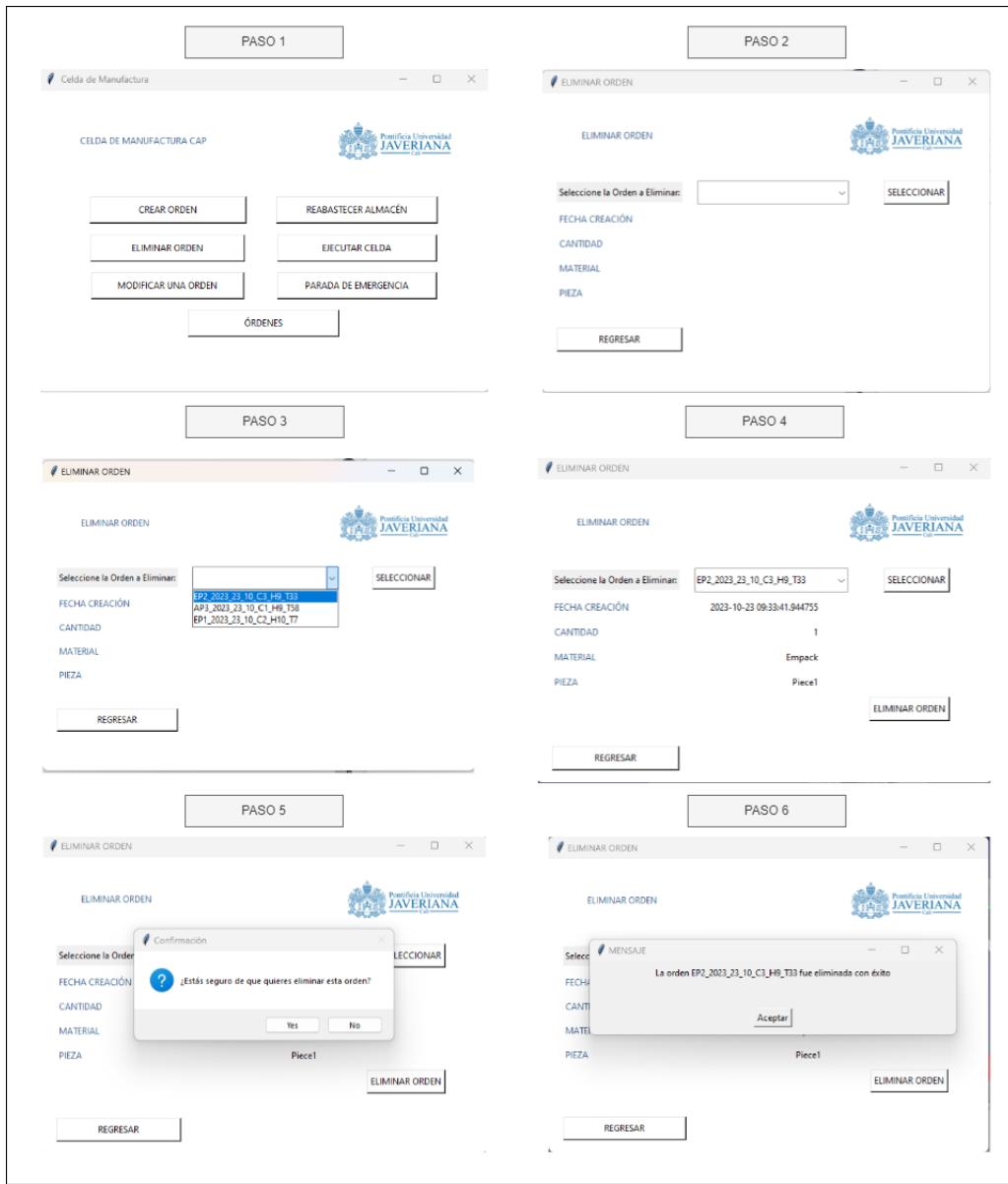


Figura 8.29: Secuencia de pasos para Eliminar una Orden

Al eliminar la orden, automáticamente la base y el Dashboard son actualizados:

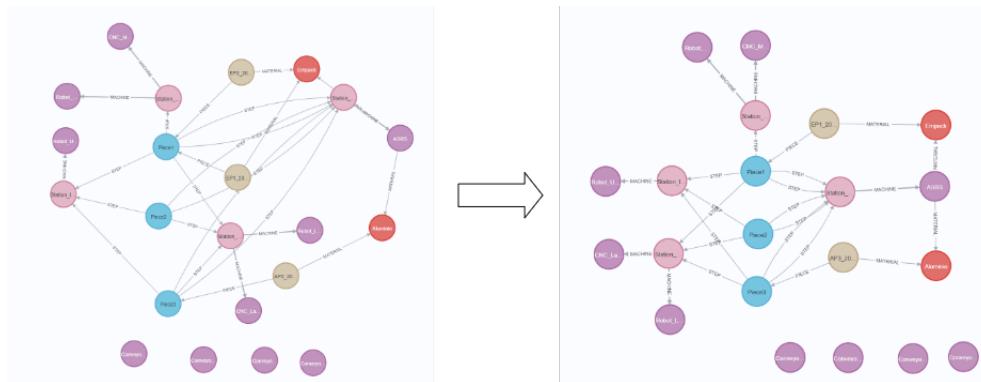


Figura 8.30: Actualización Base de Datos al Eliminar Orden

En la figura 8.30, se visualiza como el nodo correspondiente a la orden con número de ID **EP2_2023_10_C3_H9_T33** y su información se eliminan exitosamente. Como la orden ocupaba 1 materia prima de Empack, el nodo Empack se actualiza. La actualización del nodo se aprecia en la siguiente figura 8.31:

Node properties >	
material	
<id>	29
Available	12
Create_Date	9/23/2023
Location	[[1, 4], [1, 5], [2, 5], [3, 3], [3, 4], [3, 5], [4, 3], [4, 4], [4, 5], [5, 3], [5, 4], [5, 5]]
Name	Empack
Update_Date	2023-10-23 11:51:43.141360
Used	[[1, 3], [2, 3], [2, 4]]

Node properties >	
material	
<id>	29
Available	13
Create_Date	9/23/2023
Location	[[1, 3], [1, 4], [1, 5], [2, 5], [3, 3], [3, 4], [3, 5], [4, 3], [4, 4], [4, 5], [5, 3], [5, 4], [5, 5]]
Name	Empack
Update_Date	2023-10-23 13:30:46.852090
Used	[[2, 3], [2, 4]]

Figura 8.31: Actualización Material Empack

En referencia al Dashboard, a continuación en la figura 8.32 se ve la actualización correspondiente al eliminar la orden.

Para acceder al vídeo de la prueba **Eliminar Orden**, ir al siguiente link: [Prueba Eliminar Orden](#)

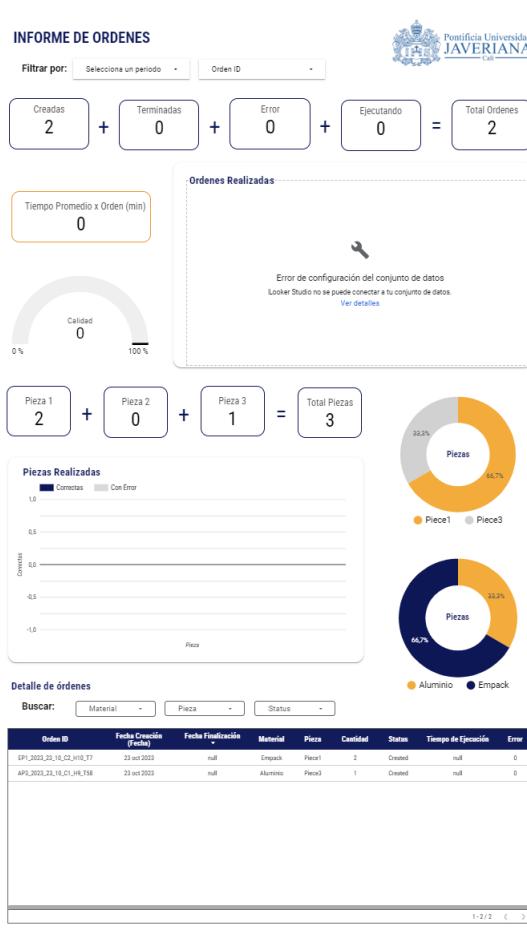


Figura 8.32: Actualización Dashboard al Eliminar Orden

8.6.4. Pruebas Botón Ver Órdenes

Para ejecutar la prueba de ver Órdenes, se deben de seguir los siguientes pasos:

- **Paso 1:** Damos clic al botón **Órdenes**, este nos dirigirá a una nueva ventana para ver las órdenes.
- **Paso 2:** Se abre la nueva ventana de Órdenes con la información en formato tabla, de las órdenes que actualmente existen en la base de datos. Cabe resaltar que la información que brinda es: ID, fecha de creación, material, cantidad y estatus.

En la figura 8.33, se muestra visualmente los pasos que se deben de ejecutar para crear una orden.

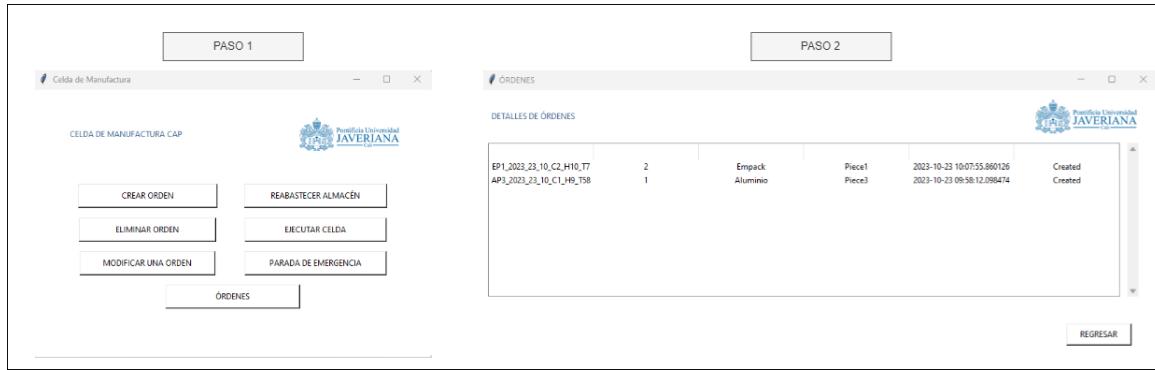


Figura 8.33: Secuencia de pasos para ver las Órdenes

Para acceder al vídeo de la prueba **Ver Órdenes**, ir al siguiente link: [Prueba Ver Órdenes](#).

Al validar el conjunto de pruebas realizadas, se concluyó que se logró establecer una buena comunicación entre los diferentes módulos desarrollados. Cabe aclarar, que las pruebas de los botones **Ejecutar celda** y **Re-establecer Almacén** se dejaron para la etapa de Operar, ya que son las que más influyen en la manipulación de la Celda de Manufactura del CAP.

Operación del Sistema de Información

9.1. Pruebas Botón Activar Celda de Manufactura

Para ejecutar la prueba de **Activar Celda**, se tomará el caso del ID correspondiente a la orden **:EP1_2023_6_11_C1_H8_T51**. Dado esto, los pasos a seguir son los siguientes:

- **Paso 1:** Damos clic al botón **Activar Celda**, donde aparecerá un mensaje de advertencia confirmando si está seguro de proceder con la activación de la celda.
- **Paso 2:** Al confirmar la activación de la celda, automáticamente la pantalla de inicio se bloquea, para dar paso a la ejecución de la celda.
- **Paso 3:** Al haber ejecutado todas las órdenes, aparecerá una nueva ventana, indicando que la celda se ha ejecutado exitosamente. De lo contrario, indicará que no existe ninguna orden que se deba de ejecutar.

En la figura 9.1, se muestra visualmente los pasos que se deben de ejecutar para activar la celda.

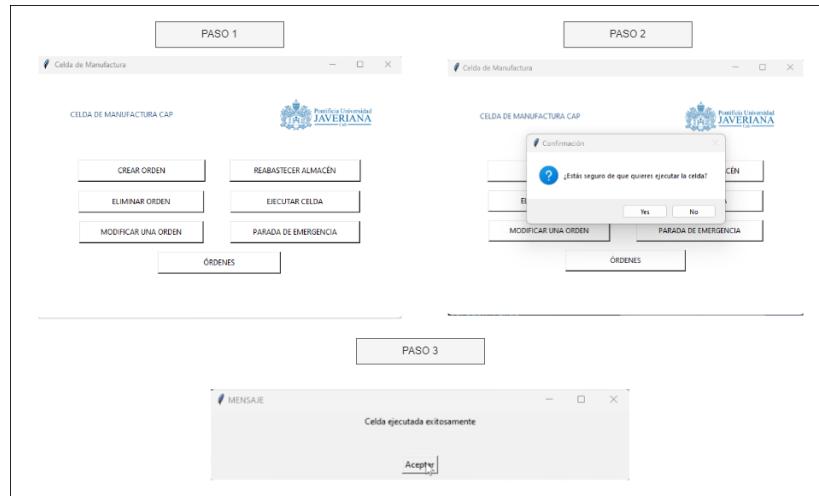


Figura 9.1: Secuencia de pasos para Ejecutar Celda

Al ejecutar la celda, automáticamente la base y el Dashboard son actualizados:

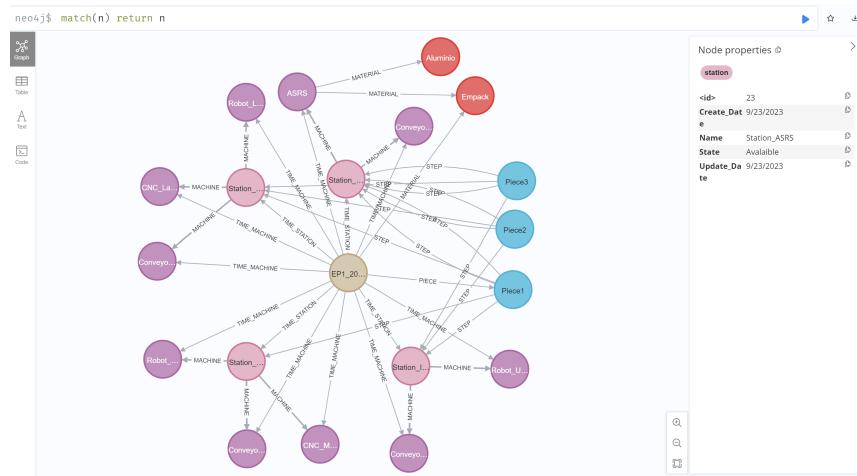


Figura 9.2: Actualización Base de Datos

Como se puede apreciar en la figura 9.2, se generaron nuevas relaciones apuntando tanto a las estaciones como a las máquinas. Estas relaciones son: **TIME_MACHINE** y **TIME_STATION**, referentes al tiempo que se demoró ejecutando la orden en la estación y máquina en específico.

Estas relaciones son de gran importancia, ya que a partir de esa información se generará la mayoría de indicadores industriales seleccionados.



Figura 9.3: Actualización de piezas producidas

Como podemos ver en la figura 9.3, se actualiza instantáneamente la cantidad de piezas producidas, en este caso, la **Pieza 1**.

En cuanto al dashboard, a partir de los cálculos de los indicadores industriales, se obtuvo el siguiente resultado:



Figura 9.4: Actualización Dashboard Órdenes



Figura 9.5: Actualización Dashboard Estación Torno



Figura 9.6: Actualización Dashboard Estación Fresado



Figura 9.7: Actualización Dashboard Estación Inspección



Figura 9.8: Actualización Dashboard Estación ASRS

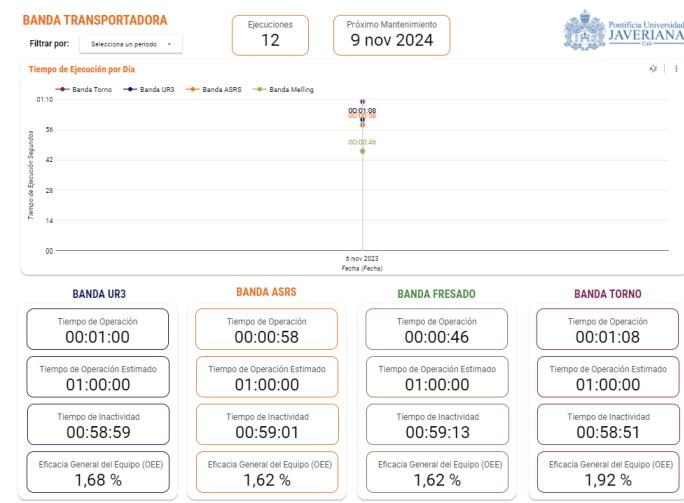


Figura 9.9: Actualización Dashboard Bandas Transportadoras

Para acceder al vídeo de la prueba **Ejecutar Celda**, ir al siguiente link: [Prueba Ejecutar Celda](#)

9.2. Pruebas Botón Re-abastecer Almacén

Para ejecutar la prueba de **Re-abastecer Almacén**, se debe de tener en cuenta los siguientes pasos:

- **Paso 1:** Damos clic al botón **Re-abastecer almacén**, donde aparecerá un mensaje de advertencia confirmando si está seguro de proceder con la acción.
- **Paso 2:** Al confirmar, automáticamente la base de datos hace un refill de la materia prima disponible, y se actualiza la simulación de RoboDK, agregando nuevas materias primas y guardando aquellas que ya se han ejecutado.

En la figura 9.10, se muestra visualmente los pasos que se deben de ejecutar para reabastecer el almacén.



Figura 9.10: Secuencia de pasos para Re-abastecer Almacén

Al re-abastecer el almacén, automáticamente la base de datos se actualiza:

Node properties >		
material		
<id>	31	⊕
Available	10	⊕
Create_Date	9/23/2023	⊕
Location	[[1,1],[1,2],[2,1],[2,2],[3,1],[3,2], [4,1],[4,2],[5,1],[5,2]]	⊕
Name	Aluminio	⊕
Update_Date	2023-11-07 19:57:25.072656	⊕
Used	□	⊕

Figura 9.11: Actualización Material Aluminio

Node properties >		
material		
<id>	30	⊕
Available	15	⊕
Create_Date	9/23/2023	⊕
Location	[[1,3],[1,4],[1,5],[2,3],[2,4],[2,5], [3,3],[3,4],[3,5],[4,3],[4,4],[4,5], [5,3],[5,4],[5,5]]	⊕
Name	Empack	⊕
Update_Date	2023-11-07 19:57:22.196250	⊕
Used	□	⊕

Figura 9.12: Actualización Material Empack

Como podemos ver en las figuras 9.11 y 9.12, los nodos tipo material (Empack y Aluminio), vuelven a re-establecerse, es decir, vuelve a tener disponibilidad en el almacén y por ende, de las ubicaciones en donde se encuentra el material.

En cuanto a la simulación en RoboDK, esta se re-inicia ubicando de nuevo toda la materia prima en sus posiciones iniciales, y guardando aquellas que ya se han ejecutado, para brindarle espacio a las nuevas que se van a ejecutar. Este cambio lo podemos visualizar en la figura 9.13 a continuación:

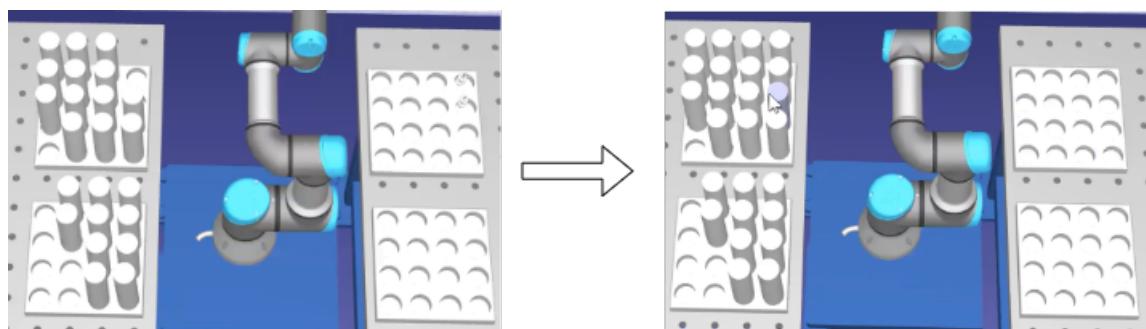


Figura 9.13: Re-abastecer Almacén RoboDK

Para acceder al vídeo de la prueba **Re-abastecer Almacén**, ir al siguiente link: [Prueba Re-establecer Almacén](#)

CAPÍTULO 10

Dificultades

La implementación del proyecto de grado se enfrentó a desafíos significativos debido al estado actual de las máquinas y estaciones en la Celda de Manufactura del CAP. Estas dificultades se originaron principalmente en las limitaciones tecnológicas y operativas de las máquinas físicas presentes en el entorno de fabricación tales como desgastes en las correas de la banda transportadora y en el ASRS, daño en uno de los controladores de los ejes del almacén y problemas en la batería de la máquina CNC del torno. Lo anterior, impedía la realización de pruebas rigurosas y sistemáticas en un entorno físico de ejecución de manufactura debido a la inoperabilidad de la celda. A continuación, se muestran imágenes de las dificultades presentadas:

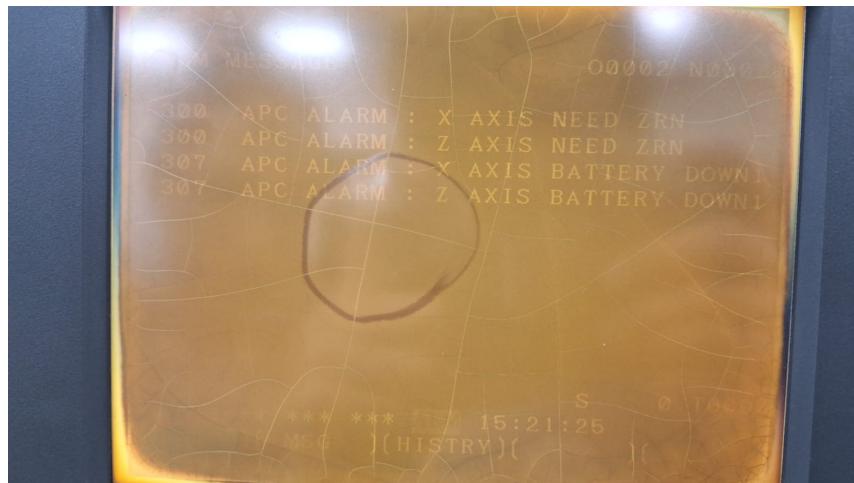


Figura 10.1: Daño en la batería del Torno CNC

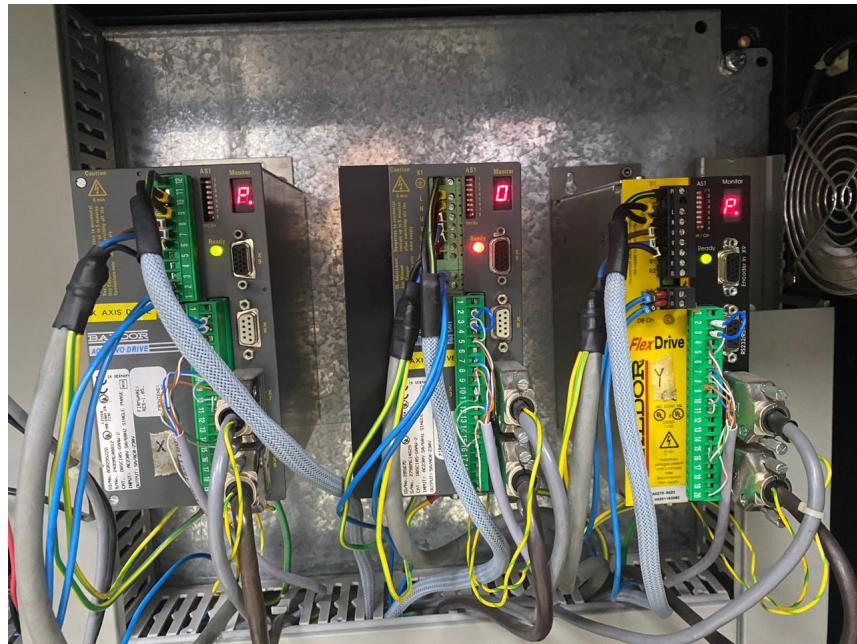


Figura 10.2: Daño en el controlador del eje Z del ASRS

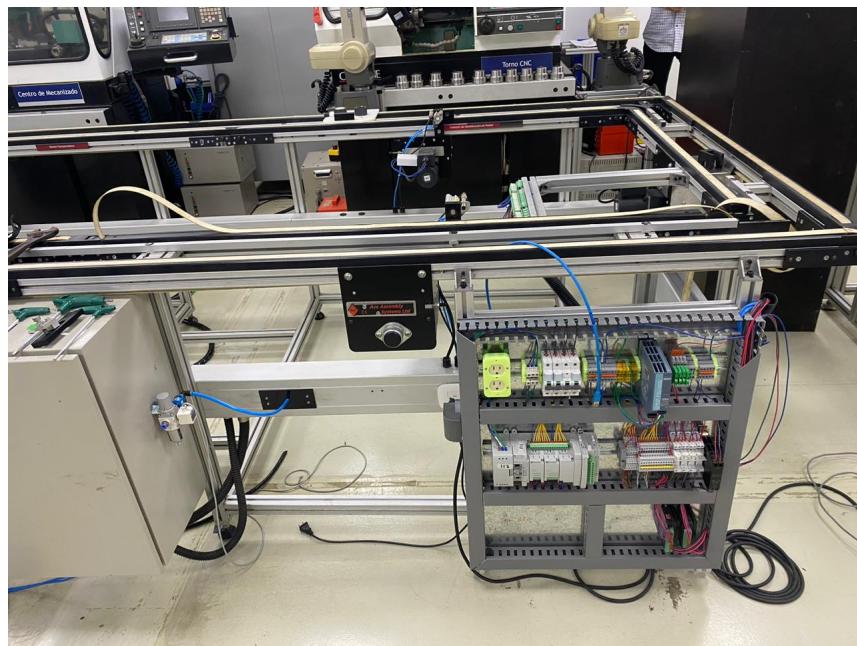


Figura 10.3: Daño en la correa de la banda transportadora



Figura 10.4: Daño en la correa de la banda transportadora



Figura 10.5: Daño en la correa del ASRS

En respuesta a esto, se tomó la decisión de simular la celda de manufactura en el software de simulación y programación para robots industriales, RoboDK, con el fin de asegurar la continuidad del proyecto a pesar de las restricciones impuestas por las condiciones del entorno físico. La simulación en RoboDK permitía una reproducción fiel de la lógica de la celda, la implementación de algoritmos y la realización de pruebas sin los riesgos asociados con las máquinas físicas.

Es importante señalar que la decisión de simular la celda de manufactura se tomó después de una evaluación y consulta con los directores del proyecto, los evaluadores y el director de carrera, por lo que implementarla en RoboDK fue considerada como una solución viable y prudente para avanzar en el proyecto, permitiendo la continuidad de la investigación y el desarrollo sin comprometer la calidad ni la integridad del trabajo.

Conclusiones y Trabajos Futuros

11.1. Conclusiones

Este proyecto de grado ha alcanzado sus objetivos establecidos, demostrando tanto la viabilidad como la capacidad para implementar un sistema de información robusto y efectivo para la Celda de Manufactura del CAP. A través de la superación de desafíos tecnológicos y operativos, como fueron las limitaciones en las máquinas físicas presentes en la celda, hemos logrado desarrollar un entorno de simulación robusto y versátil en RoboDK. Esta elección estratégica nos ha permitido no solo superar las limitaciones iniciales, sino también crear un ambiente de simulación flexible que puede adaptarse a diversas configuraciones y condiciones.

El desarrollo del ambiente de simulación en RoboDK no solo ha sido un logro técnico, sino también una herramienta poderosa que puede representar cualquier sistema de manufactura real, proporcionando una plataforma para realizar pruebas rigurosas, explorar diferentes escenarios y optimizar procesos. Además, la flexibilidad de esta simulación se ha convertido en un recurso relevante para futuras investigaciones y mejoras en el ámbito de la manufactura.

En este orden de ideas, las principales conclusiones de este proyecto son las siguientes:

1. La implementación de un sistema de información orientado a grafos en una celda de manufactura flexible puede mejorar la valorización de la información recolectada, el control y la gestión de los datos, y otorgar flexibilidad en cuanto a los cambios repentinos del proceso de fabricación. La capacidad de manejar datos y realizar consultas ad hoc ha resultado ser una ventaja significativa.
2. La implementación de un sistema de información robusto para la Celda de Manufactura del CAP es viable y puede establecer una base sólida para la optimización y automatización futura de procesos de fabricación similares.
3. El uso de una base de datos orientada a grafos puede facilitar la formulación de consultas complejas y es una opción muy viable al representar la estructura y comportamiento de la Celda de Manufactura Flexible de manera topográfica.
4. La implementación de un sistema de información robusto para la Celda de Manufactura del CAP, no sólo mejora la eficiencia y la eficacia del proceso de fabricación, sino que también puede reducir los costos y aumentar la calidad del producto final. La base sólida establecida por

este proyecto de grado puede ser utilizada como punto de partida para futuras investigaciones y mejoras en la automatización y optimización de procesos de fabricación similares.

11.2. Trabajos futuros

A pesar del éxito alcanzado en este proyecto, existen áreas y posibilidades de mejora que podrían explorarse en trabajos futuros. Las áreas clave identificadas para la expansión y el perfeccionamiento del sistema son las siguientes:

- a. **Pruebas en Entornos Reales y otros sistemas MES:** La validación y prueba del sistema en entornos reales, potencialmente en otras celdas de manufactura con diferentes características, permitirá evaluar su rendimiento en condiciones de trabajo genuinas y adaptarlo según sea necesario para satisfacer las demandas específicas de diferentes entornos de manufactura.
- b. **Inversión en la Celda de Manufactura:** Explorar la posibilidad de inversiones en la infraestructura de la celda de manufactura actual para abordar las limitaciones tecnológicas y operativas que inicialmente llevaron a la decisión de optar por una simulación.
 - **Mejoras en el Sistema del ASRS:** Actualizar el sistema del Almacén Automatizado (ASRS) para la detección temprana de fallas y para gestionar la disponibilidad del material de manera más eficiente, incluyendo la posibilidad de expansión del alcance del ASRS y acople de sensores en las secciones de este.
 - **Optimización de la Comunicación:** Investigar tecnologías de comunicación avanzadas como el Internet de las cosas industrial (IIoT) y Ethernet para mejorar la interconexión y la comunicación entre los equipos, permitiendo una gestión más ágil y en tiempo real. Además, de tener en cuenta la conexión que permite Neo4j con Looker Studio para tener una mayor capacidad de analítica y eficiencia en la gestión de los datos.
- c. **Mejoras en la Interfaz de Usuario (GUI):** Considerar adaptaciones en la interfaz de usuario para proporcionar información más detallada sobre el estado y el progreso de la celda de manufactura. Priorizar las órdenes en función de varios contextos, como la disponibilidad de material y los tiempos estimados, para una gestión más eficaz.
- d. **Desarrollo de Gemelos Digitales para el sistema MES:** La creación de gemelos digitales, puede permitir un mejor análisis y optimización avanzada de los procesos de manufactura al emular el comportamiento de la celda, ofreciendo así información en tiempo real para la toma de decisiones ante los cambios en las órdenes demandadas.
- e. **Implementación de Modelos de Inteligencia Artificial (IA):** Utilizar los datos recopilados en la base para desarrollar modelos de IA que puedan colaborar en la predicción de posibles fallas en las máquinas con el fin de proponer estrategias de mantenimiento preventivo y optimizar el uso de los recursos disponibles para la gestión de inventarios, permitiendo un seguimiento de los materiales y productos en la cadena de suministro, beneficiando así la eficiencia logística y la reducción de costos.

CAPÍTULO 12

Anexos

- Diseño Base de Datos
- Interfáz Gráfica GUI
- Simulación RoboDK
- Software de Control
- Script de consultas a Base de Datos
- Script de comunicación a Dashboard
- Prueba Crear Orden
- Prueba Modificar Orden
- Prueba Eliminar Orden
- Prueba Ver Órdenes
- Prueba Ejecutar Celda
- Prueba Re-establecer Almacén
- Dashboard
- Repositorio de Github

Bibliografía

- [1] V. Salazar, “Análisis de la integración de los sistemas mes–erp en industrias de manufactura,” in *Energy and technology for the Americas: education, innovation, technology and practice. Latin American and Caribbean Conference for Engineering and Technology, San Cristobal, Venezuela*, 2009, pp. 1–6.
- [2] J. R. Capacho and W. Nieto Bernal, *Diseño de bases de datos*. Universidad del Norte, 2017.
- [3] M. J. Antiñanco, “Bases de datos nosql: Escalabilidad y alta disponibilidad a través de patrones de diseño,” Ph.D. dissertation, Universidad Nacional de La Plata, 2014.
- [4] TigerGraph, “What is a graph database and why should you care?” Jul 2019. [Online]. Available: <https://medium.com/@tigergraph/what-is-a-graph-database-and-why-should-you-care-d537124c7f39>
- [5] L. Calabuig Monerris, “Bases de datos orientadas a grafos aplicadas al estudio de informes radiológicos: utilizando entornos de computación en la nube para abordar estudios de gran dimensión,” Sep 2016. [Online]. Available: <http://hdl.handle.net/10251/64172>
- [6] N. Francis, A. Green, P. Guagliardo, L. Libkin, T. Lindaaker, V. Marsault, S. Plantikow, M. Rydberg, P. Selmer, and A. Taylor, “Cypher: An evolving query language for property graphs,” in *Proceedings of the 2018 International Conference on Management of Data*, ser. SIGMOD ’18. New York, NY, USA: Association for Computing Machinery, 2018, p. 1433–1445. [Online]. Available: <https://doi.org/10.1145/3183713.3190657>
- [7] M. Hermann, T. Pentek, B. Otto *et al.*, “Design principles for industrie 4.0 scenarios: a literature review,” *Technische Universität Dortmund, Dortmund*, vol. 45, 2015.
- [8] C. Zhang, G. Zhou, Y. Jing, R. Wang, and F. Chang, “A digital twin-based automatic programming method for adaptive control of manufacturing cells,” *IEEE Access*, vol. 10, pp. 80 784–80 793, 2022.
- [9] M. M. Maran, N. A. Paniavin, and I. A. Poliushkin, “Alternative approaches to data storing and processing,” in *2020 V International Conference on Information Technologies in Engineering Education (Inforino)*, 2020, pp. 1–4.
- [10] P. Shi, G. Fan, S. Li, and D. Kou, “Big data storage technology for smart distribution grid based on neo4j graph database,” in *2021 IEEE 4th International Conference on Electronics Technology (ICET)*, 2021, pp. 441–445.
- [11] M. C. Pabón Burbano, “Lenguaje visual de consulta basado en transformación de grafos: aplicación en el dominio médico,” 2016.

- [12] T. A. Undheim, "The future of manufacturing execution systems is simplicity," Jun 2022. [Online]. Available: <https://www.forbes.com/sites/trondarneundheim/2022/06/07/the-future-of-manufacturing-execution-systems-is-simplicity/?sh=401caf7b6fb9>
- [13] M. Hany, Y. Liu, K. Montgomery, and R. Candell, "Wireless cyber-physical systems performance evaluation through a graph database approach," 2020-10-01 00:10:00 2020. [Online]. Available: https://tsapps.nist.gov/publication/get_pdf.cfm?pub_id=929563
- [14] F. Tao, Q. Qi, A. Liu, and A. Kusiak, "Data-driven smart manufacturing," *Journal of Manufacturing Systems*, vol. 48, pp. 157–169, 2018, special Issue on Smart Manufacturing. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S0278612518300062>
- [15] C. H. dos Santos, G. T. Gabriel, J. V. S. do Amaral, J. A. B. Montevechi, and J. A. de Queiroz, "Decision-making in a fast fashion company in the industry 4.0 era: a digital twin proposal to support operational planning." *International Journal of Advanced Manufacturing Technology*, vol. 116, no. 5/6, pp. 1653 – 1666, 2021. [Online]. Available: <https://search-ebscohost-com.bdbib.javerianacali.edu.co/login.aspx?direct=true&db=a9h&AN=151760487&lang=es&site=ehost-live>
- [16] J. Qiao, A. Zhou, and H. Qiu, "Research and implementation of system of electric business expanding installation process based on graph database," in *2020 IEEE 3rd International Conference on Automation, Electronics and Electrical Engineering (AUTEEE)*, 2020, pp. 10–13.
- [17] S. Timón-Reina, M. Rincón, and R. Martínez-Tomás, "Overview of graph databases and their applications in the biomedical domain." *Database: The Journal of Biological Databases Curation*, vol. 2021, pp. 1 – 22, 2021. [Online]. Available: <https://search-ebscohost-com.bdbib.javerianacali.edu.co/login.aspx?direct=true&db=a9h&AN=155028445&lang=es&site=ehost-live>
- [18] M. Tisch, C. Hertle, J. Cachay, E. Abele, J. Metternich, and R. Tenberg, "A systematic approach on developing action-oriented, competency-based learning factories," *Procedia CIRP*, vol. 7, pp. 580–585, 2013, forty Sixth CIRP Conference on Manufacturing Systems 2013. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S2212827113003053>
- [19] L. Kohnová and N. Salajová, "Impact of industry 4.0 on companies: Value chain model analysis," Jan 2023. [Online]. Available: <https://www.mdpi.com/2076-3387/13/2/35>
- [20] F. d. Ingenieria, "Revista ingeniería investigación y tecnología: Facultad de ingeniería," 2019. [Online]. Available: <https://www.revistaingenieria.unam.mx/numeros/v20n4-05.php>
- [21] B. Bidanda, P. Ariyawongrat, K. L. Needy, B. A. Norman, and W. Tharmmaphornphilas, "Human related issues in manufacturing cell design, implementation, and operation: a review and survey," *Computers Industrial Engineering*, vol. 48, no. 3, pp. 507–523, 2005, groupTechnology/Cellular Manufacturing. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S0360835204001998>

- [22] N. I. of Standards and Technology, “Guide to industrial control systems security,” May 2015. [Online]. Available: <https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-82r2.pdf>
- [23] M. Mecalux, “Software de trazabilidad: Todos los productos bajo control,” Mar 2020. [Online]. Available: <https://www.mecalux.com.mx/blog/software-trazabilidad>
- [24] “Reglamento (ce) no 178/2002 del parlamento europeo y del consejo de 28 de enero de 2002,” Jan 2002. [Online]. Available: <https://eur-lex.europa.eu/legal-content/ES/TXT/PDF/?uri=CELEX:32002R0178&from=ES>
- [25] Vertech, “9 mes tools every plant manager needs to improve business performance,” 2017. [Online]. Available: <https://www.vertech.com/hubfs/White%20Papers/9%20MES%20Tools.pdf?hsCtaTracking=f14e0820-3ca4-4135-8e8f-b453214858e5%7C0be8bff7-5607-4381-ac7f-f688e98b0948>
- [26] Oracle, “¿qué es una base de datos relacional (sistema de gestión de bases de datos relacionales)?” 2022. [Online]. Available: <https://www.oracle.com/co/database/what-is-a-relational-database/>
- [27] A. W. Services, “¿qué es una base de datos relacional?” 2023. [Online]. Available: <https://aws.amazon.com/es relational-database/>
- [28] J. Sánchez, “Principios sobre bases de datos relacionales,” *Informe, Creative Commons*, vol. 11, p. 20, 2004.
- [29] H. G. del Busto and O. Y. Enríquez, “Bases de datos nosql,” *Telemática*, vol. 11, no. 3, pp. 21–33, 2012.
- [30] A. C. Romero, J. S. G. Sanabria, and M. C. Cuervo, “Utilidad y funcionamiento de las bases de datos nosql,” *Facultad de Ingeniería*, vol. 21, no. 33, pp. 21–32, 2012.
- [31] V. Sharma and M. Dave, “Sql and nosql databases,” *International Journal of Advanced Research in Computer Science and Software Engineering*, vol. 2, no. 8, 2012.
- [32] IONOS, “Graph database: Bases de datos para una interconexión eficiente,” Oct 2019. [Online]. Available: <https://www.ionos.es/digitalguide/hosting/cuestiones-tecnicas/graph-database/#:~:text=Uno%20de%20los%20puntos%20fuertes,velocidad%20incluso%20en%20b%C3%A1squedas%20complicadas>
- [33] I. Neo4j, “Clause composition - cypher manual.” [Online]. Available: https://neo4j.com/docs/cypher-manual/current/clauses/clause_composition/
- [34] O. Foundation, “Opc unified architecture,” Sep 2019. [Online]. Available: <https://opcfoundation.org/about/opc-technologies/opc-ua/>

- [35] ——, “Opc classic,” Oct 2020. [Online]. Available: <https://opcfoundation.org/about/opc-technologies/opc-classic/>
- [36] Paessler, “Monitor your industrial environment using opc ua,” Jun 2021. [Online]. Available: <https://www.paessler.com/opc-ua-monitoring>
- [37] S. AG, “Using a cp 443-1 opc ua as gateway for mes/erp,” May 2017. [Online]. Available: https://cache.industry.siemens.com/dl/files/832/109743832/att_923351/v3/109743832_CP443-1_OPc_UA_V14_DOKU_V10_en.pdf
- [38] S. M. Person and I. INC, “What is opc ua and how does it affect your world?” May 2008. [Online]. Available: <https://www.plantengineering.com/articles/what-is-opc-ua-and-how-does-it-affect-your-world/>
- [39] I. E. COMMISSION, “Iec 62541-5 - international electrotechnical commission,” Jun 2020. [Online]. Available: https://webstore.iec.ch/preview/info_iec62541-5%7Bed3.0.RLV%7Den.pdf
- [40] Z. Lin and S. Pearson, “An inside look at industrial ethernet communication protocols,” Jul 2018. [Online]. Available: <https://www.ti.com/lit/wp/spry254b/spry254b.pdf>
- [41] A. INCORPORATED, “Introduction to modbus tcp/ip - prosoft technology,” 2005. [Online]. Available: https://www.prosoft-technology.com/kb/assets/intro_modbustcp.pdf
- [42] E. P. COMPANY, “Industrial ethernet communication protocols,” 2019. [Online]. Available: https://www.encoder.com/hubfs/white-papers/WP-2019_Industrial-Ethernet-Protocols/wp2019-industrial-ethernet-communication-protocols.pdf
- [43] A. Andersdotter, D. J. Lansford, D. Law, D. R. B. Marks, and D. A. Myles, “Ieee 802 standards association,” May 2023. [Online]. Available: <https://standards.ieee.org/featured/ieee-802/>
- [44] M. Organization, “Modbus messaging on tcp/ip implementation guide v1,” Oct 2006. [Online]. Available: https://modbus.org/docs/Modbus_Messaging_Implementation_Guide_V1_0b.pdf
- [45] T. Sharma, “Rs232 serial communication protocol: Basics, working amp; specifications,” Jan 2018. [Online]. Available: <https://circuitdigest.com/article/rs232-serial-communication-protocol-basics-specifications>
- [46] P. Horowitz and W. Hill, *The Art of Electronics*, 2nd ed. Cambridge University Press, 1989.
- [47] D. L. User, “Computer integrated manufacturing systems (cim’s) equipment installation supplement,” Sep 2002. [Online]. Available: <https://www.kleineducational.com/manufacturers/denford-ltd>

- [48] B. Kan, W. Zhu, G. Liu, X. Chen, D. Shi, and W. Yu, “Topology modeling and analysis of a power grid network using a graph database,” *International Journal of Computational Intelligence Systems*, vol. 10, p. 1355, 01 2017.
- [49] M. Liu, X. Li, J. Li, Y. Liu, B. Zhou, and J. Bao, “A knowledge graph-based data representation approach for iiot-enabled cognitive manufacturing,” *Advanced Engineering Informatics*, vol. 51, p. 101515, 2022. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S1474034621002639>