EINDHOVEN UNIVERSITY OF TECHNOLOGY

Department of Mathematics and Computer Science
Process Analytics

Big Data Management and Analytics (BDMA)

Analysis of Resource Working Behaviour and Working Patterns Utilizing High-Level Batch Aggregation

Master Thesis

Liliia Aliakberova

Supervisors:
Dirk Fahland

25-08-2024

Abstract

This thesis examines the analysis of resource behavior and working patterns using high-level batch aggregation in a European company (referred to as Company C), which specializes in sterilizing medical equipment. The research addresses the question: What patterns in employee work habits and task prioritization can be identified within Company C's sterilization process, and how can these patterns define the features provided for a digital twin that enhances operational efficiency and compliance? To address this, we developed a methodology that first identifies batching within the Event Knowledge Graph of Company C. These batches are then aggregated into high-level events to reflect iterative patterns. We further analyze resource behavior and working patterns based on this high-level event log. The findings reveal how processing is influenced by workload and highlight common working patterns across weekdays and shifts. These insights are intended to support the development of a digital twin that enhances operational efficiency and compliance.

Keywords: Resource Behavior, Working Patterns, Batching, High-Level Batch Aggregation, Event Knowledge Graph (EKG), Process Mining

Preface

As I reflect on this journey, my first thoughts are with my parents. I would like to express my deepest gratitude to my mother and father. They have been my heroes, showing me what success, hard work, and persistence look like. Their unwavering support has been always there for me despite the distance. Their love and encouragement have guided me every step of the way. Without them, none of this would have been possible.

My partner, Rick, has been by my side through all the ups and downs. His belief in me, especially when I doubted myself, has given me strength. Your presence in my life has been invaluable, Rick, and I am endlessly grateful for it. I also want to extend my thanks to Rick's family for their kindness and support throughout this journey.

To my friends, both those I have known for years and the ones I have met in the BDMA program—thank you for your support and companionship. Your friendship has made this journey so much richer and more enjoyable.

I am deeply grateful to the BDMA consortium for the unique opportunity to study at three distinguished universities. This experience has allowed me to explore Europe, immerse myself in different cultures, and learn new languages. For me, the BDMA program was not just about earning a degree; it was a transformative adventure that expanded my horizons in countless ways.

I also want to express my sincere appreciation to the professors for sharing their knowledge and guiding me during my master degree. A special thanks to Professor Dirk Fahland for his guidance, assistance, and for providing both constructive and critical feedback that has been essential in shaping this work.

Lastly, I want to thank myself for not giving up.

Contents

1 Introduction **6**
- 1.1 Context and Topic . . . 6
- 1.2 State of the Art . . . 7
- 1.3 Research Questions . . . 9
- 1.4 Research Method . . . 10
- 1.5 Findings . . . 11

2 Background **14**
- 2.1 AUTO-TWIN Project Description . . . 14
- 2.2 Business Understanding . . . 15
  - 2.2.1 Introduction to Company C . . . 15
  - 2.2.2 Sterilization Center Arrangement . . . 15
  - 2.2.3 The Sterilization Process . . . 17
- 2.3 Initial Data Understanding . . . 18
  - 2.3.1 Initial Data Understanding: Data Issues . . . 20
  - 2.3.2 Initial Data Understanding: Resource Perspective . . . 21
- 2.4 Problem statements from the business perspective . . . 23

3 Discussion on Preliminaries and Related Work **24**
- 3.1 Process Mining . . . 24
- 3.2 Graphs . . . 26
  - 3.2.1 Labeled Property Graphs (LPGs) . . . 26
  - 3.2.2 Neo4j and the Cypher Query Language . . . 27
- 3.3 Multi-dimensional Process Mining . . . 28
- 3.4 Resource-Oriented Process Mining . . . 30
  - 3.4.1 Batching . . . 30
  - 3.4.2 Human Resource Working Patterns . . . 33
- 3.5 Analytical Techniques in Process and Data-Driven Research: Correlation and Sequence Analysis . . . 33
  - 3.5.1 Normalization and Correlation Analysis . . . 34
  - 3.5.2 Frequent Sequence Analysis . . . 34

4 Problem Exposition **36**
4.1 Detailed Research Questions 36
4.2 Detailed Method 37

5 From EKG Implementation to Extended Resource Exploration and Task Instances Analysis **42**
5.1 EKG Implementation 43
5.1.1 Preprocessing for EKG Implementation 43
5.1.2 EKG Implementation Statistics 43
5.2 Extended Resource Exploration Utilizing EKG 44
5.3 Applying Task Instance Framework over Company C's EKG . 47
5.3.1 Limitations of the Approach 47
5.3.2 Need for Batch Detection Revision 48

6 Batching **49**
6.1 Batching in the Event Knowledge Graph: Proposing Options for Company C 49
6.2 Batching over Resource 53
6.2.1 Mathematical Definition of Batching over Resource . . 55
6.2.2 Batching over Resource: Implementation Steps 56
6.2.3 Batching over Resource: Implementation Results and Exploration 59
6.3 Batching over Activity 65
6.3.1 Mathematical Definition of Batching over Activity . . . 67
6.3.2 Batching over Activity: Implementation Steps 67
6.3.3 Batching over Activity: Implementation Results and Exploration 72

7 High-Level Batching **80**
7.1 Motivation and Requirements for High-Level Batch Aggregation 80
7.1.1 Batching Patterns that Lead to High-Level Aggregation 83
7.1.2 Requirements for High-level Batching Aggregation . . . 85
7.2 High-Level Batching: Definitions and Implementation Steps . 86
7.2.1 Mathematical Definition of High-Level Batches 86
7.2.2 Implementation of High-Level Batching Aggregation . . 87
7.2.3 High-Level Batching: Implementation Results and Representation in the EKG 89

8 High-Level Resource Behavioral Analysis **94**
8.1 Forming High-Level Event Log 94

8.2 Resource Working Behavior Based on High-Level Batching and Workload . . . 96
8.2.1 Defining the Workload . . . 97
8.2.2 Detailed Approach for Analyzing Resource Working Behavior . . . 97
8.2.3 Results Evaluation of Analyzing Resource Working Behavior . . . 102
8.2.4 General Overview of Resource Working Behavior . . . 107
8.3 Frequent Working Patterns Using High-Level Aggregation . . . 108
8.3.1 Detailed Approach for Analyzing Frequent Working Patterns . . . 108
8.3.2 Results Evaluation of Frequent Working Patterns Analysis . . . 109

9 Conclusion **114**
9.1 Limitations . . . 115
9.2 Future Work . . . 116

APPENDICES **121**

A Frequent Working Patterns Analysis Results **125**

Chapter 1

Introduction

This chapter outlines the overall scope of the master thesis, which was conducted as part of the Auto-Twin project focused on developing advanced digital twin technologies. Within this project, we concentrate on the process of a European company (referred to as Company C) that specializes in sterilizing medical equipment. This chapter introduces the overall scope of the master thesis. Section 1.1 presents the main topics and provides context for the study. In Section 1.2, we review relevant literature and methodologies in the field. The central research question, which investigates identifiable patterns in employee work habits and task prioritization within Company C's sterilization process, is discussed in Section 1.3, along with subquestions that further explore how these patterns can inform the development of an effective digital twin to enhance operational efficiency and compliance. The methodology used to investigate these questions, including data exploration, the development of batching techniques, high-level batching aggregation and quantitative analysis, is detailed in Section 1.4. Finally, Section 1.5 presents the findings and their interpretations.

1.1 Context and Topic

In healthcare, the sterilization of medical devices is a process that ensures instruments are safe for use in medical procedures. This process relies not only on advanced equipment and strict protocols but also on the actions and decisions of the people involved. The effectiveness of the sterilization process depends significantly on how human resources, such as technicians and operators, interact with equipment, navigate through workstations, and coordinate with one another. Understanding this resource behavior, which includes how employees prioritize tasks and manage workloads, is important

for improving the efficiency and reliability of the sterilization process.

Recognizing the important role of human resources in these operations, the AUTO-TWIN project focuses on advancing the development and application of Digital Twins that represent digital models that replicate real-world processes. By accurately representing both the physical workflow and the behavior of the people involved, Digital Twins offer a method for optimizing business operations. One of the key areas of focus within the AUTO-TWIN project is the sterilization process at Company C, a provider of sterilization services for medical devices. In this context, the project aims to develop a Digital Twin that not only reflects the physical aspects of the sterilization process but also captures and analyzes the behavior of human resources, which is necessary for improving efficiency, reliability, and compliance.

Company C operates sterilization centers where human resources must work together with advanced equipment to ensure that medical instruments are properly sterilized. However, the complexity of these operations means that having advanced equipment and protocols alone is not sufficient. The success of the sterilization process also depends on human resources. Thus, Company C aims to better understand how its human resources work within the sterilization process in order to further develop a Digital Twin that accurately reflects these operations.

This research directly contributes to Company C's objectives by developing a framework that aggregates operational data into high-level features based on resource timelines. These features are designed to capture and analyze how human resources at Company C interact with the sterilization process over time, with a focus on identifying specific behaviors and patterns. By providing these insights, the research supports the creation of a Digital Twin. By accomplishing this, the research not only contributes to Company C but also addresses a broader gap in understanding how human resources interact with complex processes.

1.2 State of the Art

Analyzing resource behavior is a complex task, especially in environments where multiple entities interact, like equipment, personnel and stations. This complexity is particularly significant when trying to understand how employees prioritize tasks and perform their work. Process mining is a field that helps address these challenges by analyzing event logs from information systems to discover, monitor, and improve real processes [28]. By examining this event data, process mining provides insights into how processes function, how resources are allocated, and how different components of a process

interact.

One important aspect of resource behavior is the detection of patterns like batching, where similar tasks are grouped together to increase efficiency. Batching is a well-known concept in operational processes. For example, the Piled Execution pattern introduced by van der Aalst [23] is closely related to batching. Additionally, studies by Wen et al. [32] and Martin et al. [18] have expanded the understanding of batch processing in workflows and event logs. Recognizing and analyzing these batching patterns is important for understanding how resources organize and perform tasks.

Traditional process mining techniques often focus on analyzing event logs from a single entity, which may not capture the full complexity of resource behavior. This is where multi-dimensional process mining and Event Knowledge Graphs (EKGs) become important. Multidimensional process mining goes beyond single-entity analysis by examining events that involve multiple interconnected entities [5, 7]. EKGs are a powerful tool in this context, as they represent and analyze the relationships within complex, multidimensional event data [8]. By organizing events, entities, and their connections in a graph structure, EKGs allow for a more detailed analysis of processes that involve multiple actors and interactions.

Resource-oriented process mining builds on these concepts by focusing specifically on how human resources interact with and impact business processes. Instead of just tracking tasks, this approach seeks to understand the roles, interactions, and behaviors of the resources involved. This is crucial for capturing the subtleties of resource behavior, which traditional process mining methods might miss. Research by Klijn, Mannhardt, and Fahland has been key in developing frameworks that use Event Knowledge Graphs (EKGs) in multidimensional process mining to study resource behavior [15, 16]. The work on Task Instances offers a method for grouping tasks into Task Instances, which can reveal patterns in how resources work within individual cases and further generalize them into resource high-level events across the cases that a resource has participated in.

However, even with these advances, there are still challenges in fully understanding and modeling resource behavior in complex, multi-entity environments. Additionally, traditional batching techniques may not sufficiently represent the iterative and complex work patterns typical in such environments.

In this thesis, we aim to address these challenges by proposing new batching detection techniques with time constraints, specifically within the context of multidimensional process mining and Event Knowledge Graphs. Furthermore, we introduce a high-level batch aggregation approach that helps to capture and analyze these complex workflows more effectively. This combination

of approaches will enable a more comprehensive analysis of resource behavior, particularly in identifying and understanding working patterns across multiple cases simultaneously.

1.3 Research Questions

Building on the existing frameworks and techniques discussed in the previous Section 1.2, the primary objective of this thesis is to develop a framework that aggregates operational data into high-level features based on resource timelines. This framework is designed to capture user actions, offering an overview of resource activities over time. These high-level features, focusing on specific employee behaviors and patterns, are important for constructing a digital twin that aims to provide digital representation of real-world sterilization process in Company C.

Given this main goal, the following research question and subquestions have been formulated:

Main research question: What are the identifiable patterns in employee work habits and task prioritization within Company C's sterilization process, and how can these patterns define the features provided for a digital twin that enhances operational efficiency and compliance?

Research subquestions:

- Can the Task Instances framework from Klijn, Mannhardt, and Fahland's work [15, 16] be effectively applied to represent batching processing in the sterilization process and aid in identifying and analyzing the working patterns of resources within Company C?

- What other batching detection techniques, can be employed to aggregate resource workflow for identifying and analysing resource behavior in the sterilization process?

- Are existing batching techniques sufficient for capturing the complex work behaviors of resources, or is there a need for additional aggregation methods to better identify and analyze work patterns?

- How do employees prioritize tasks within the sterilization process, and what factors influence their prioritization decisions?

- Identify working patterns that could serve as indicators for features that explain work for further modeling in a digital twin.

1.4 Research Method

The approach employed in this thesis was created to address the main research question and subquestions. The approach integrates several main stages:.

1. The research commenced with an initial data exploration phase, detailed in Section 2.3 of Chapter 2, where key operational patterns such as batching and data issues were identified. Subsequently, we conducted data preprocessing and implemented the Event Knowledge Graph, as discussed in Section 5.1 of Chapter 5. Additionally, we extended the exploration to include shift-based resource allocation, as outlined in Section 5.2. This stage was crucial for establishing the research direction and identifying gaps in existing process analysis techniques relevant to the AUTO-Twin project.

2. Faced with the limitations of the standard Task Instance Framework in capturing the nuanced behaviors of resources within the Company C's Event Knowledge Graph described in the Secton 5.3, two new batch detection techniques were conceptualized and developed: Batching over Resource and Batching over Activity with temporal constraints in the Chapter 6. The latter was selected for deeper analysis due to its effectiveness in defining and integrating group work, thereby providing a more compact and representative technique.

3. To address the shortcomings of selected batching technique in capturing iterative work patterns, a high-level aggregation approach was introduced in the Chapter 7. This involved categorizing work patterns into regular, iterative, and complex iterative batches

4. Utilizing the refined Event Knowledge Graph, the study progressed to a quantitative analysis phase in the Chapter 8, where the impact of workload on process throughput was systematically examined across different stages of the sterilization process described in the Section 8.2. This analysis leveraged high-level aggregated logs to explore how workload at various stages influences processing.

5. The final part of the analysis involved employing frequent sequence mining techniques to uncover common working patterns across different shifts, and weekdays described in the Section 8.3. This phase aimed to identify and understand the collaborative and operational dynamics that characterize the sterilization process.

The results of this analysis contribute to the development of a digital twin for Company C, defining high-level features that capture resource workload behavior and working patterns, thereby enhancing the digital representation of real-world sterilization process in Company C.

The code presented in the GitHub repository reflects the implementation of this methodology. While the actual data and Event Knowledge Graph used in this study are not included due to confidentiality agreements.

In general, this thesis is organized into nine chapters, each addressing a specific aspect of the research. Chapter 1 provides an overview of the study. Chapter 2 introduces the necessary background information, covering the AUTO-TWIN project, Company C's operations, and the initial data understanding. Chapter 3 discusses foundational concepts and related work, focusing on process mining, graphs, multi-dimensional process mining, and relevant resource analytical techniques. Chapter 4 details the research questions and methodology used to tackle the identified challenges. Chapter 5 describes the implementation of the Event Knowledge Graph (EKG) and extended resource analysis related to the shift work structure, highlighting the need for revising batch detection methods. Chapter 6 explores batching techniques within the EKG, focusing on resources and activities, and presents the implementation results. Chapter 7 introduces high-level batch aggregation, detailing its requirements and implementation within the EKG. Chapter 8 analyzes resource behavior using high-level batching and workload data, identifying working patterns and evaluating the findings. Finally, Chapter 9 summarizes the research outcomes, discusses limitations, and suggests directions for future work.

1.5 Findings

This section presents the key findings from the analysis of resource behavior and work patterns within the sterilization process at Company C. The research aimed to identify how resources interact with tasks and prioritize their work. The following subsections will discuss the results obtained from the study and provide an interpretation of these findings in the context of providing high-level features for developing the digital twin of Company C.

Results

This thesis analyzed resource behavior and work patterns within the sterilization process at Company C using. The research began with an exploration of data, which revealed key patterns such as *batching* and *shift-based*

GitHub Repository: https://github.com/Liliia-Aliakberova/ThesisResearchHighLevelBatchAggregation

work in the sterilization workflow. Further, we proposed *two batching detection techniques Batching over Resource* and *Batching over Activity* utilizing *Event Knowledge graph* of the company. Batching over Activity was chosen because it effectively defined collaborative work and provided a clearer analytical approach. However, this method alone did not capture the repetitive work patterns seen among resources. To address this, *high-level aggregation techniques* were used to group work patterns into regular, iterative, and complex iterative categories, allowing for a more organized analysis of resource behavior.

The study then used a *high-level aggregation log* constructed from high-level batches, based on the EKG, to examine how *workload*, measured as the number of kits waiting to be processed, impacted the processing within sterilization process. The findings showed that workload impacts not only the processing of kits but also influences the workload on later stages, highlighting the interconnectedness of the workflow. Additionally, frequent sequence mining was used to find *common work patterns* across different weekdays and shifts, uncovering distinct collaborative behaviors and operational details within shift structures.

Interpretation

The results of this research provide important insights into how resources are allocated and how tasks are performed in complex operations like the sterilization process at Company C. Building on the work of Wen et al. [32], Martin et al. [18], and Klijn, Mannhardt, and Fahland [15, 16], the newly developed batch detection techniques and high-level aggregation methods proved effective in capturing and analyzing complex resource behaviors that were previously challenging to model. The study's findings highlight the significant impact of workload on the efficiency of the sterilization process, particularly how it impacts the processing on different stages of the workflow.

Moreover, identifying common work patterns across shifts and weekdays helps in better understand how resources collaborate and operate under different conditions. These insights not only improve the understanding of current operations at Company C but also support future developments in multi-dimensional process mining and the creation of high-level resource features for digital twins.

Importantly, the observed correlations between high-level batches and workload, along with the detection of frequent working patterns, demonstrate that these high-level batches are not just relevant to this specific case but offer a valuable concept for process analysis more broadly. By contributing to the existing knowledge in process mining techniques, this research introduces

a tool that can be applied to various complex operations, thereby enhancing the analytical capabilities within the field of process mining.

In summary, this thesis addresses the challenges of analyzing resource behavior in a sterilization context and provides a foundation for future advancements in multi-dimensional process mining, offering valuable direction for developing digital twins to optimize operational efficiency.

Chapter 2

Background

This chapter outlines the background context for this master's thesis. It starts with an overview of the AUTO-TWIN project in the Section 2.1, establishing the basis for the research. Following this, we describe Company C, its functions and operations. In Section 2.2, we explain the sterilization process at Company C, which is the central subject of this thesis. Section 2.3 offers a preliminary overview of the data utilized for analysis. Lastly, Section 2.4 addresses the problem statements from a business perspective, clarifying the importance of this thesis. Together, these sections provide the necessary background and context to understand the behavior of Company C's resources in the sterilization process.

2.1 AUTO-TWIN Project Description

The AUTO-TWIN project [1] tackles important technological and economic challenges. Its main goal is to improve the development and use of Digital Twins [24], making business operations more efficient, reliable, and sustainable within circular economies [25]. The project introduces several advanced methods and technologies:

- **Digital Twins**: AUTO-TWIN uses an automated, process-aware approach to create Digital Twins [24]. These Digital Twins accurately replicate real-world operations digitally to support reliable business processes [1].

- **IDS-based Common Data Space**: The project uses an International Data Space (IDS). A data space is a virtual environment that offers a standardized framework for data exchange, utilizing common protocols

and formats [21]. This improves data interoperability and resource use in a circular-economy ecosystem [25].

- **Smart Green Gateways**: A Green Gateway is a strategic decision-making tool that helps maximize the value and efficiency of products and materials [10]. AUTO-TWIN integrates advanced hardware into the digital thread to create smart Green Gateways that make efficient and environmentally friendly data-driven decisions using Digital Twins.

The AUTO-TWIN project includes three specific use cases, one of which is the Sterilization Process in the Health Sector at Company C. This thesis focuses on this use case.

The effectiveness of the sterilization process at Company C largely depends on human resources allocation and behavior, which are key factors in maintaining an efficient workflow. Therefore, this thesis focuses on understanding human resource behavior and working patterns in this process. This understanding is essential for developing a Digital Twin with AUTO-TWIN project that accurately reflects the real-world process and improves operational performance and sustainability, aligning with the project's goals.

2.2 Business Understanding

2.2.1 Introduction to Company C

Company C has established itself as a company responsible for distribution and administration of surgical instruments and medical equipment. With many years of experience, the company has adapted to meet the needs of the healthcare industry. It offers a range of products and services aimed at maintaining high standards of medical care. One important contribution of Company C is the development of advanced sterilization centers.

2.2.2 Sterilization Center Arrangement

This thesis examines an sterilization process at the advanced sterilization center of Company C inside of a hospital. Figure 2.1 shows the center's design, which supports an efficient workflow with dedicated rooms and areas. Each area has specific machines and tools for different stages of the sterilization process.

The sterilization process commences with acceptance of medical devices in the sterilization center through the hallway. The first area they encounter is the dirty area, where the devices, assembled into medical kits, unpacked

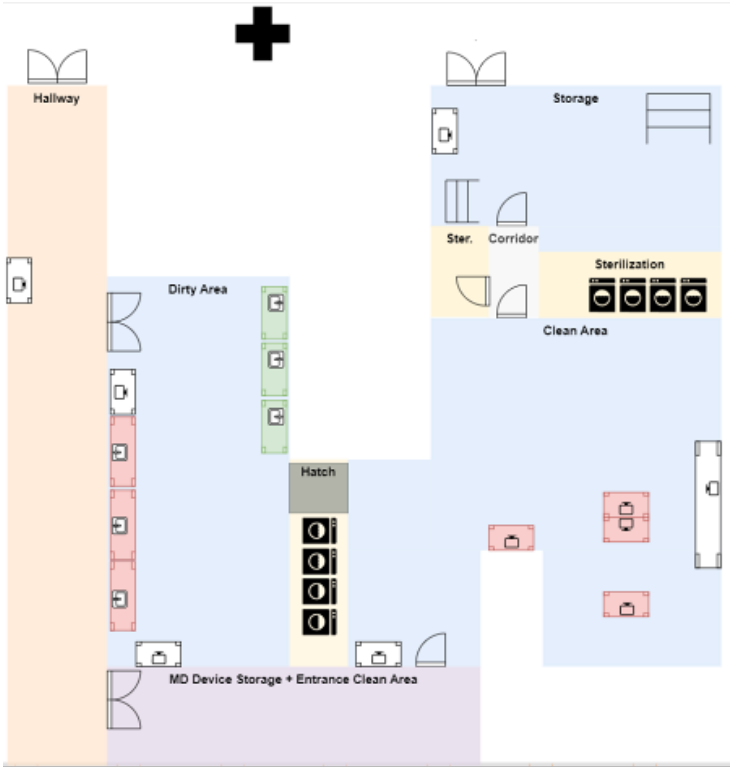

Figure 2.1: Facility Arrangement of Company C's Sterilization Center

and pre-washed. This area is essential for removing any contaminants before the kits move to more specialized cleaning processes. Metal kits are placed into automated washing machines that are designed for high-throughput and efficient cleaning, while plastic kits are manually cleaned to prevent damage due to their material properties.

After the initial cleaning phase, both types of kits proceed to the medical device storage and entrance cleaning area for a detailed inspection. This area serves as a critical checkpoint where kits are examined for any remaining contaminants or damage. Metal kits are then packed in preparation for sterilization, while plastic kits are unpacked to ensure they are ready for the next phase.

Once the devices pass inspection, they move to the clean area. This area is designed for further cleaning processes to ensure the devices are free from any contaminants. Following this, the devices proceed to the sterilization area, where they undergo the sterilization process. The sterilization area is equipped with multiple sterilizers to handle a high volume of devices efficiently.

After sterilization, the devices are transferred to the storage area. This zone is used to keep the sterilized devices in a controlled environment until they are needed for use.

Throughout the process, scanners are used at various stations to scan the kits as they go through each step of the sterilization process, ensuring proper tracking and documentation. The facility also includes a sterilization corridor that connects different areas of the center, ensuring a smooth transition of devices from one zone to another. Additionally, there is a hatch used for passing items between areas while maintaining the cleanliness and sterility of the environment.

Each part of the sterilization center is specifically designed to support a streamlined and effective sterilization process, ensuring that medical devices are handled safely and efficiently from entry to storage.

2.2.3 The Sterilization Process

The next important step is to examine the sterilization process in detail. The process presents is a multi-stage operation designed to ensure the highest level of cleanliness and sterility for medical devices, including both plastic and metal kits. The Figure 2.2 represents the C company sterilization process.

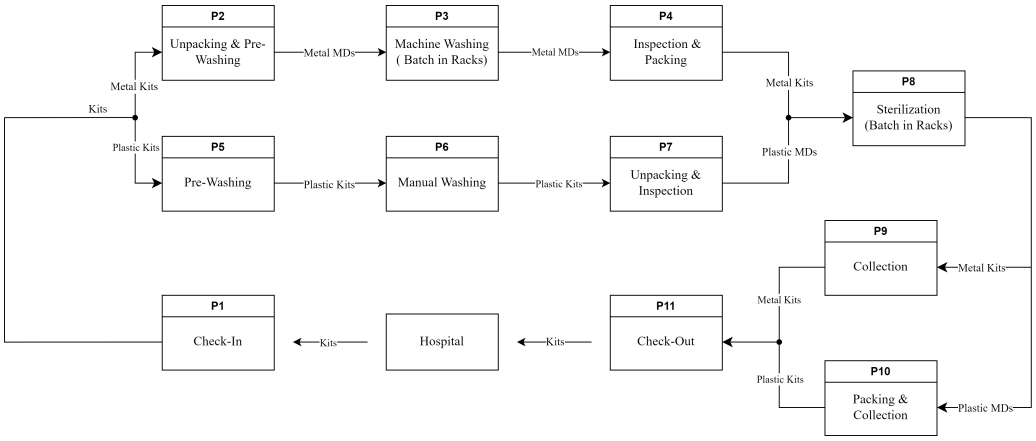

Figure 2.2: Company C Sterilization Process Diagram

The process begins with a Check-in phase (P1) for both plastic and metal kits, where the kits are logged and prepared for the subsequent stages. For metal kits, the process continues with unpacking and pre-washing (P2), followed by machine washing (P3) using racks for batch processing. This method ensures that multiple kits can be processed simultaneously, increasing efficiency. After machine washing, metal kits undergo inspection and

packing (P4), where they are thoroughly checked and repacked for sterilization.

Plastic kits, on the other hand, follow a slightly different path due to their material properties. They undergo pre-washing (P5), manual washing (P6), and unpacking and inspection (P7). Manual washing is particularly important for plastic kits to prevent any potential damage from automated machines. During the inspection phase, both types of kits are checked for any malfunctions or remaining contaminants.

Once the initial cleaning and inspection stages are completed, both types of kits converge for the sterilization stage (P8) using racks for batch processing. This stage is critical as it involves the actual sterilization process, where the kits are exposed to high temperatures and pressure to be fully sterilized.

After sterilization, the process diverges again. Metal kits move to the collection stage (P9), where they are prepared for storage. This involves final inspections and packaging to ensure they remain sterile until use. Plastic kits proceed to the packing and collection stage (P10), where they are similarly inspected and packed. Finally, all kits go through the Checkout stage (P11), marking the completion of the sterilization process. This stage involves final logging.

In addition, the process involves several human and physical resources. There are 20 unique human resources involved in the sterilization process at Company C, each playing an important role in ensuring the smooth operation of the workflow. They are responsible for scanning barcodes and ensuring compliance with the First In, First Out (FIFO) principle, which ensures that kits are sterilized in the order they arrive. This practice maintains the order and traceability of the kits throughout the entire process.

Overall, the sterilization center at Company C exemplifies a well-organized and efficient system designed to meet the highest standards of sterility and safety in the healthcare industry.

2.3 Initial Data Understanding

Another important aspect of the project context is the initial understanding of the data. Company C provided the historical process data that spans from January 1st, 2022, to March 31st, 2022, and was provided directly from their internal system. We received a sample of the complete dataset consisting of six original files in CSV format. Notably, provided dataset included the data for metal kits. Each record in the dataset corresponds to a specific event (task) performed on a kit at a particular sterilization station by a designated operator.

Below is a detailed list showing the original data column names, their English translations, and explanations for each entity:

- **Fecha de seguimiento (Monitoring Date)**: The date when the kit was monitored during the sterilization process.
- **Hora de seguimiento (Tracking Time)**: The specific time when the kit was tracked at a particular process stage.
- **Usuario (User)**: The operator responsible for handling the kit at a specific process stage.
- **Nombre punto de control (Checkpoint Name)**: The name of the activity (station) where the kit is being processed.
- **Es preliminar (Parameter of Preliminary)**: Indicates whether the process is in a preliminary stage or has priority.
- **Es fisico (Parameter of Being Physical)**: Indicates whether the process involves a physical handling stage (e.g., manual washing).
- **Es denegado (Parameter of Denial)**: Indicates whether the kit was denied at a certain checkpoint due to issues.
- **Cant. (Quantity)**: The number of medical devices in the kit processed at a particular stage.
- **Tipo de objeto (Type of a Kit)**: The type or category of the kit being processed (e.g., metal or plastic).
- **Codigo (Kit Code)**: A unique code assigned to each kit for identification.
- **N/S (Kit Sequence Number)**: The sequence number of the kit, used to track its order in the process.
- **Nombre/Descripcion (Description of a Kit)**: A detailed description of the kit, including its contents and purpose.
- **Tipo de produccion (Production Type)**: The type of production process the kit is associated with.
- **Nombre produccion (Production Name)**: The name of the specific production process.

- **Informacion adicional (Additional Information)**: Any additional information relevant to the kit or process stage.

In AUTO-TWIN project, these six initial data files were preprocessed into nine files. Each of the nine files contains data for a specific type of work performed at its corresponding Nombre punto de control (Checkpoint Name). Table 2.1 below provides information about the dataset files, the amount of events at each activity (station) and their associated process stages, as introduced in Section 2.2.3.

Table 2.1: Number of Events by Process Activity Name and Associated Process Stages

Activity Name	**Number of Events**	**Process Stage**
Entrada Material Sucio	13,537	P1
Cargado en carro L+D	19,692	P2
Carga L+D iniciada	24,738	P3
Carga L+D liberada	23,617	P3
Montaje	24,171	P4
Producción montada	37,931	P4
Composición de cargas	37,813	P8
Carga de esterilizador liberada	37,199	P9
Comisionado	22,958	P11
Total	241,565	-

2.3.1 Initial Data Understanding: Data Issues

It is important to mention that we detected several issues with the data that we consider for further processing. These issues include:

1) **Duplicated Events**: The analysis revealed a significant presence of duplicates within the provided event tables with entities. Table 2.2 illustrates the frequency of duplicates linked to each input file. Notably, these duplicates constitute approximately 10% of the entire dataset. It is important to note that we consider duplicates as errors, and we will not include them while loading the data into EKG presented in the Section 5.1.1

2) **Kits Disappearing After a Specific Sterilization Step**: Initially, we detected an anomaly where a significant portion of kit codes (based

Table 2.2: Number of Duplicates per File

Activity	**Number of Duplicates**
Entrada Material Sucio	512
Cargado en carro L+D	4,349
Carga L+D iniciada	9,617
Carga L+D liberada	9,319
Montaje	1,270
Producción montada	0
Composición de cargas	10
Carga de esterilizador liberada	21738
Comisionado	71
Total	22,637

on the Código attribute from the initial data) appeared only during a specific sterilization activity before disappearing, affecting 202 out of 1183 kits. Here we analysed codigo values that have up to 4 unique activities. However, after introducing the KitID in the analysis in the Section 5.1.1, this issue was largely resolved. Only a small portion of these kits that exhibited this disappearing behavior were not considered when loading the data into the Event Knowledge Graph (EKG). The construction of EKG is presented in the Section 5.1

2.3.2 Initial Data Understanding: Resource Perspective

From the Resource perspective we mentioned earlier that the process involves 20 unique human resources involved in the sterilization process at Company C. Resources are responsible for scanning barcodes of kits on sterilization stations and ensuring compliance with FIFO principle. FIFO in this case is related to kits that must be sterilized in the order they arrive. We provide additional insights regarding the resources below.

Distribution of Work We decided to verify the amount of work per resource from the data to better understand the workload distribution across various resources. The bar chart on the Figure 2.3 visualizes this distribution, with the x-axis representing the resources and the y-axis indicating the number of rows (events) associated with each resource.

The analysis reveals that the resource labeled "ER" handles the most significant volume of tasks, with a total of 22,032 rows. This is followed

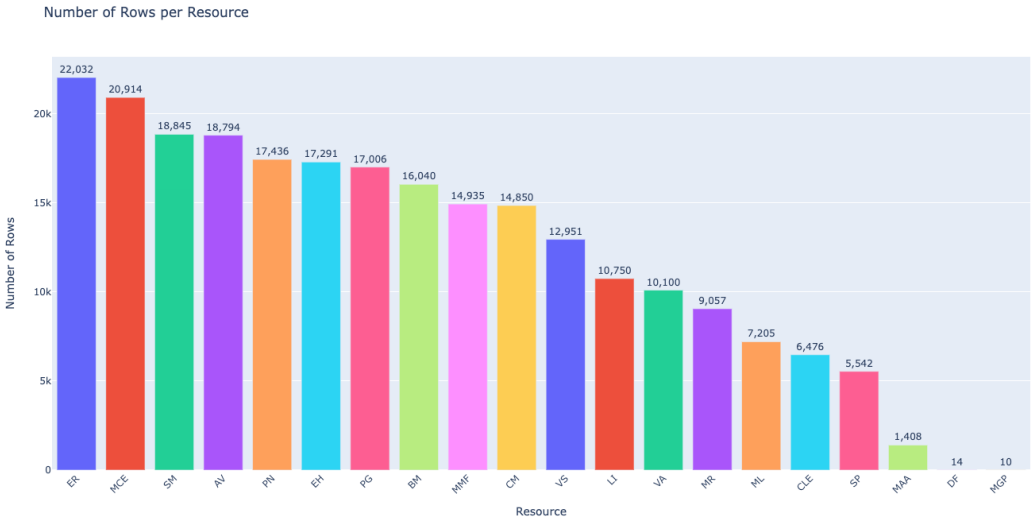

Figure 2.3: Distribution of Events (rows) for Each Resource

by "MCE" and "SM," which manage 20,914 and 18,845 rows respectively. These three resources clearly carry the most of the workload, highlighting their important roles within the process.

Resources such as "PN," "EH," and "PG," with task counts ranging from 16,000 to 17,500 rows, represent a middle tier in terms of workload. These resources appear to have a more balanced distribution of tasks, suggesting that they may operate in a collaborative manner within the process.

At the lower end of the spectrum, resources like "SP," "MAA," and particularly "DF" and "MGP" have significantly fewer tasks, with "MGP" managing only 10 rows. This sharp contrast could indicate that these resources are involved in a limited number of activities or underrepresented in the provided portion of the data.

In summary, the chart shows a high disparity in workload distribution among the resources.

Batch Processing Pattern In the initial examination of the data from a resource perspective, we identified a distinct pattern known as batching processing. This pattern occurs when a resource, represented by the "Usarion" (User) attribute, performs several similar tasks consecutively. In the dataset, this was evident as sequences of rows where both the "Usarion" (User) and "Nombre punto de control" (Checkpoint Name) attributes—indicating the resource and the specific activity they were engaged in—remained consistent across multiple entries. Despite the consistency in user and activity, the kit attributes in these rows varied, signifying that the resource was processing

different kits in the same activity.

This observation of batching behavior is important because it reveals how resources manage their tasks by grouping similar activities together. Such behavior can have significant implications for understanding working behaviour, resource allocation, and the overall workflow within the sterilization process. Recognizing and analyzing these batching patterns can help us analysis resource workflows.

2.4 Problem statements from the business perspective

In the previous section 2.3, we discussed the FIFO principle in Company C, where resources follow FIFO for kits. FIFO from kit perspective in Comapny C means that the first kits to enter a process are also the first to exit. This principle regulate the flow of medical kits through various stages of sterilization, ensuring that older kits are processed before newer ones. This helps maintain the integrity and orderliness of the sterilization process, reducing the risk of outdated or expired kits being used.

However, FIFO is implemented only on a task-specific basis and does not apply to the management of human resources. Although kits are handled sequentially, the organization of employee tasks does not follow FIFO. For example, workers might adhere to FIFO in specific tasks like unpacking or pre-washing, but this system does not necessarily influence how they move between different tasks. This inconsistency presents challenges in understanding employee task prioritization and movement, which is crucial for operational insights at Company C.This creates a complex environment that the company currently lacks insight into.

For the digital twin simulation [19], which seeks to accurately replicate real-world conditions, it is important to model the decision-making process of workers. Therefore, the simulation model must understand the criteria by which workers decide to remain at a station or switch to another. Understanding and integrating this working behavior and patterns is key to accurately mirror real-world scenarios and enhance the model's effectiveness.

Chapter 3

Discussion on Preliminaries and Related Work

This chapter provides an overview of the important concepts and related literature necessary to understand the methodologies and approaches employed in this research. It begins with a discussion on process mining in the Section 3.1. This section defines key terms such as events, event tables, and entities.

The chapter then explores the concept of graphs in the Section 3.2, particularly Labeled Property Graphs (LPGs). Additionally, we introduce Neo4j, a graph database system, and its query language that were used in this research to manipulate and query company C data.

Subsequently, the chapter describes multi-dimensional process mining in the Section 3.3, on how this concept is utilized in the AUTO-TWIN project, which uses Event Knowledge Graphs (EKGs) as a standard data format for analyzing event data.

Following previous, the next Section 3.4 covers the idea of resource-oriented process mining, which shifts the focus to understanding the roles, behaviors, and interactions of resources within business processes. Finally, the chapter reviews analytical techniques in process and data-driven research in the Section 3.5 that provides details on the methods used in this thesis to perform resource behavioral analysis.

3.1 Process Mining

In event-driven systems, specific discrete observation activities, referred to as **events**, are recorded by various means such as sensors, systems, or human observers at particular times. These events are organized into structured formats known as **event tables**.

Event tables are the main structure that organizes events by grouping them with attributes and connecting them to entities. Entities represent objects or subjects involved in the events, and the event table records the sequence of these events to help analyze and understand the relationships between them.

Definition 1. ***(Event Table with Entities):*** *From an event table perspective, events constitute a finite sequence from the event set E as $T = (E, Attr, \#, ENT)$ that represents an event table with a set $Attr$ of attribute names including $act, time \in Attr$ and entities that belong to specific entity types $ent \in ENT$ [8].*

Definition 2. ***(Events):*** *An event, denoted as $e \in E$. Each observation is encapsulated by attribute-value pairs through the partial function $\# : E \times AN \rightarrow Val$. For each event $e \in E$ and attribute name $a \in AN$, the notation $\#(e, a) = v$ specifies the value v of attribute a for event e. If attribute a is undefined for event e (i.e., it has no value), we denote it as $\#(e, a) = \perp$. Alternatively, we express $\#(e, a) = v$ as $\#_a(e) = v$ or $e.a = v$.*

For every event $e \in E$, we require:

1. The attribute time is defined, meaning $\#_{\text{time}}(e) \neq \perp$.
2. Event e possesses a non-undefined value $\#_a(e) \neq \perp$ for at least one other attribute $a \in AN$, where $a \neq$ time [6].

Entities correspond to specific types of objects or subjects within event data. For example, in the Company C sterilization process, we consider entities like kit and resource. **A Case** refers to a particular entity tracked over time based on the events it is involved in. Each run of sterilization process for a particular kit can be considered a case, with events capturing its journey through various stages of the process.

Definition 3. ***(Entities, Case, Case Identifier):*** *Based on the Definition 1 let $ent \in ENT$ denotes an entity type. The set $Entities(ent, T)$, comprising entities of type ent in T, is defined as $Entities(ent, T) = \{n | \exists e \in E : n \in e.ent\}$ [8]. We define Case as following $Case(ent, time(e))$ and $Cases(T, id)$ consists of the unique values of the entity designated as the case identifier across all events in the event table [6]. The **Case Identifier** is a unique identifier assigned to each case within the event table. If we select an attribute $id \in AN$ (from the set of attributes $Attr$) as the case identifier, then the set of cases for this identifier is given by $Cases(T, id) = \{\pi_{id}(e_i) \mid e_i \in E$, where π_{id} is the projection function that extracts the value of the attribute id for each event e_i in the event table T [6].*

Definition 4. ***(Event Log):*** *An* ***Event Log*** *is a collection of event records that groups events into cases using a* ***Case Identifier****. Events in a case are ordered as a* ***Trace****, representing one "run" of the process [30].For example, in the sterilization process of company C we can see a trace as one sterilization cycle for a kit. An event log is a multiset of traces, allowing comparison of multiple sequences. Formally, an event log L is a set of traces:*

$$L = \{\sigma_1, \sigma_2, \ldots, \sigma_m\}$$

where each trace $\sigma = \langle e_1, e_2, \ldots, e_n \rangle$ corresponds to the sequence of events in a case [6].

In summary, process mining involves analyzing event data to understand and optimize business processes.

3.2 Graphs

This thesis research uses concept of graphs.

Definition 5. ***(Graph)*** *A graph is a mathematical structure used to model pairwise relations between objects. It consists of a set of nodes (also called vertices) connected by edges (also known as links or arcs).*

Formally, a graph G is defined as an ordered pair $G = (V, E)$ [4], where:

- *V is a set of vertices, representing the objects in the graph.*
- *E is a set of edges, representing the relationships or connections between the vertices.*

3.2.1 Labeled Property Graphs (LPGs)

Labeled Property Graphs (LPGs) extend the basic graph model by introducing labels and properties to both nodes and edges. In an LPG, each node and edge can have a label [14], which categorizes the type of node or edge, and a set of key-value pairs known as properties, which store additional information about the node or edge.

Definition 6. ***(Labeled Property Graphs)*** *Formally, a labeled property graph G can be defined as:*

$$G = (V, E, \mathcal{L}_V, \mathcal{L}_E, \mathcal{P}_V, \mathcal{P}_E)$$

where:

- V *is a set of vertices (nodes).*
- E *is a set of edges (relationships).*
- $\mathcal{L}_V$ *is a labeling function that assigns labels to vertices.*
- $\mathcal{L}_E$ *is a labeling function that assigns labels to edges.*
- $\mathcal{P}_V$ *is a property function that assigns key-value pairs (properties) to vertices.*
- $\mathcal{P}_E$ *is a property function that assigns key-value pairs (properties) to edges.*

LPGs provide the structure for the Event Knowledge Graphs used in this thesis and which will be introduced in the next Section 3.3

3.2.2 Neo4j and the Cypher Query Language

Neo4j is a graph database management system [22, 13] that implements the labeled property graph model. It is designed to store, manage, and query large-scale graph data efficiently.

Neo4j uses Cypher query language that is a declarative graph query language [9] that allows users to describe patterns in graphs and retrieve data from them. In Cypher, a basic query involves specifying a pattern of nodes and relationships to match in the graph [20]. For example, the following Cypher query retrieves all nodes labeled `User` and their connections to `Stations` nodes they work on:

```
MATCH (a:User)-[r:WORKS]->(b:Station)
RETURN a, r, b
```

This query finds all instances in the graph where a `User` node is connected to `Station` node through a `WORKS` relationship and returns the nodes and their connecting relationships.

In this thesis, we store and manage Company C's data in Neo4j, using Cypher to run queries on it. Neo4j's structure is well-suited for the multi-dimensional process mining we perform introduced in the following section.

3.3 Multi-dimensional Process Mining

Multidimensional process mining expands on traditional process mining techniques by considering event data that spans multiple interconnected entities, rather than relying on a single case identifier to divide the data into independent sequences [5, 8, 7]. Traditional process mining often falls short when applied to complex scenarios involving multiple related entities, as it only provides a limited view [8]. In contrast, multidimensional process mining allows for a more comprehensive analysis by examining the interactions between various entities within the event data [8].

Within this domain, the Event Knowledge Graph (EKG) is a powerful tool for representing and analyzing process data across multiple dimensions of entities. EKGs provide a flexible and structured approach to capture the complex relationships and interactions in multidimensional event data, enabling more insightful process analysis [8].

Within the AUTO-TWIN project, the Event Knowledge Graph (EKG) is adopted as the standard data format for managing and analyzing multidimensional event data. The EKG facilitates an understanding of sterilization process data, supporting the systematic development of CROMA's digital twin

Event Knowledge Graphs (EKG) are graphical representations of event data, where nodes represent events or entities, and edges represent correlation or directly-follows relationships. EKGs provide a visual and structural framework for analyzing event data, facilitating pattern discovery, anomaly detection, and process optimization. Moreover, EKGs are extensible, for instance allowing for the addition of new layers to aggregate events into higher-level events [16] or infer missing values [26]. Notably, EKGs have been effectively used to model and analyze how human actors work across multiple cases, making them particularly valuable when we want to study the user behaviour in process data [16, 8].

Definition 7. ***(Event Knowledge Graph (EKG)):*** *An event knowledge graph (or simply graph) is an LPG, where* $G = (N, R, \lambda, \#)$ *with node labels* $\{Event, Entity\} \subseteq \Lambda_N$ *and relationship labels* $\{df, corr\} \subseteq \Lambda_R$*, indicating "directly-follows" and "correlation" with the following properties:*

1. Every event node $e \in N_{\text{Event}}$ captures an activity name $e.act \neq \perp$ and a timestamp $e.time \neq \perp$.

2. Each entity node $n \in N_{\text{Entity}}$ possesses an entity type $n.type \neq \perp$.

3. Each correlation relationship $r \in R_{\text{corr}}$, denoted as $\vec{r} = (e, n)$, is defined from an event node to an entity node, where $e \in N_{\text{Event}}$ and $n \in N_{\text{Entity}}$; we use $n \in \text{corr}(e)$ and $e \in \text{corr}(n)$ as shorthand.

4. Any directly-follows relationship $df \in R_{df}$, represented as $\vec{df} = (e_1, e_2)$, is established between event nodes $e_1, e_2 \in N_{\text{Event}}$ and refers to a specific entity $df.ent = n \in N_{\text{Entity}}$, satisfying:

   (a) Both e_1 and e_2 are correlated to entity n: $(e_1, n), (e_2, n) \in R_{\text{corr}}$.

   (b) Event e_1 precedes e_2: $e_1.time < e_2.time$.

   (c) There exists no other event $e' \in N_{\text{Event}}$ correlated to n, such that $(e', n) \in R_{\text{corr}}$, occurring between $e_1.time < e'.time < e_2.time$.

   We denote $df.type = df.ent.type$ and $(e_1, e_2) \in R_{df}$ [8].

Correlation Relationships establish connections between events and entities, indicating which events are associated with which entities. These relationships enable the representation of dependencies and interactions between events and entities within the event data.

Definition 8. ***(Correlation Relationship):*** *Let $T = (E, Attr, \#, ENT)$ denote an event table with entities. Let $n \in Entities(ent, T)$ be an entity of type $ent \in ENT$. Event e is correlated to entity n, denoted as $(e, n) \in corr(ent, T)$, if either $n = e.ent$ or $n \in e.ent$. We denote $corr(n, ent, T) = \{e \in E | (e, n) \in corr(ent, T)\}$ as the set of events that correlated to entity $n \in Entities(ent, T)$ [8].*

Directly-Follows Relationships signify the temporal ordering of events for specific entities. They indicate which events directly precede others in sequence, providing insights into the temporal dynamics of processes involving those entities.

Definition 9. ***(Directly-Follows Relationship):*** *Let $T = (E, Attr, \#, ENT)$ represent an event table with entities. Let $n \in Entities(ent, T)$ be an entity of type $ent \in ENT$. Let $e_1, e_2 \in E$ be two events. Event e_2 directly follows event e_1 from the perspective of entity n, defined as $e_1 n, T e_2$, if both e_1 and e_2 are correlated to n, $e_1.time < e_2.time$, and there is no other event $(e', n) \in corr(ent, T)$ such that $e_1.time < e'.time < e_2.time$ [8].*

To illustrate Event Knowledge Graph we present Figure 3.1 of the sterilization process in the company C. The graph features correlation relationships (CORR) highlighted in green, along with directly-follows relationships

(DF_KIT, DF_RESOURCE, DF_RUN) distinguished by three different colors: pink, blue, and orange, corresponding to the associated entities Kit, Resource, and Run, respectively. In this representation, the Run entity serves as a case. Event nodes are depicted in a light brown color. In this specific EKG, correlation relationships (CORR) connect six Resource entity nodes (PG, EH, VS, ER, PN) displayed in light blue with event nodes, and event nodes connect with two entity nodes in pink labeled as both Run and Kit. The upper entity node corresponds to the Kit HNB-NC.012-1.0 with Run HNB-NC.012-1.0-2, while the lower entity node corresponds to the Kit HUBU-TR.002-1.0 and Run HUBU-TR.002-1.0-9.

Creating Event Knowledge Graphs (EKGs) involves importing event table records as nodes, inferring entities and correlations from event attributes, and ordering events by time for each entity to add Directly-Follows relations that show the sequence of events. This process can be automated using a metadata file to describe entities and their relationships [27]. In the AUTO-TWIN project for Company C, this automated approach, based on Object-Centric Event Data Models in Event Knowledge Graphs, is used to transform legacy data into OCED (Object-Centric Event Data) compliant graphs. This ensures that the data structure meets the standards required for comprehensive process mining analysis. By leveraging this methodology, the AUTO-TWIN project integrates complex, multi-entity event data into a unified framework, supporting analysis and the development of CROMA's digital twin.

3.4 Resource-Oriented Process Mining

Understanding and modeling human behavior within process mining is a challenging task due to the complexity and variability of human actions [28]. This complexity is particularly evident in resource-oriented process mining, where the goal is to analyze how resources, particularly human actors, interact with and influence processes. In this thesis, we attempt to address these challenges by leveraging EKG within multi-dimensional process mining domain.

3.4.1 Batching

In Section 2.3, we identified the presence of a batching processing pattern in the dataset, where resources performed similar tasks consecutively. This observation is consistent with the broader concept of batching in resource management, where tasks are grouped together to enhance efficiency. Batching is a well-established concept, and it has been discussed extensively in

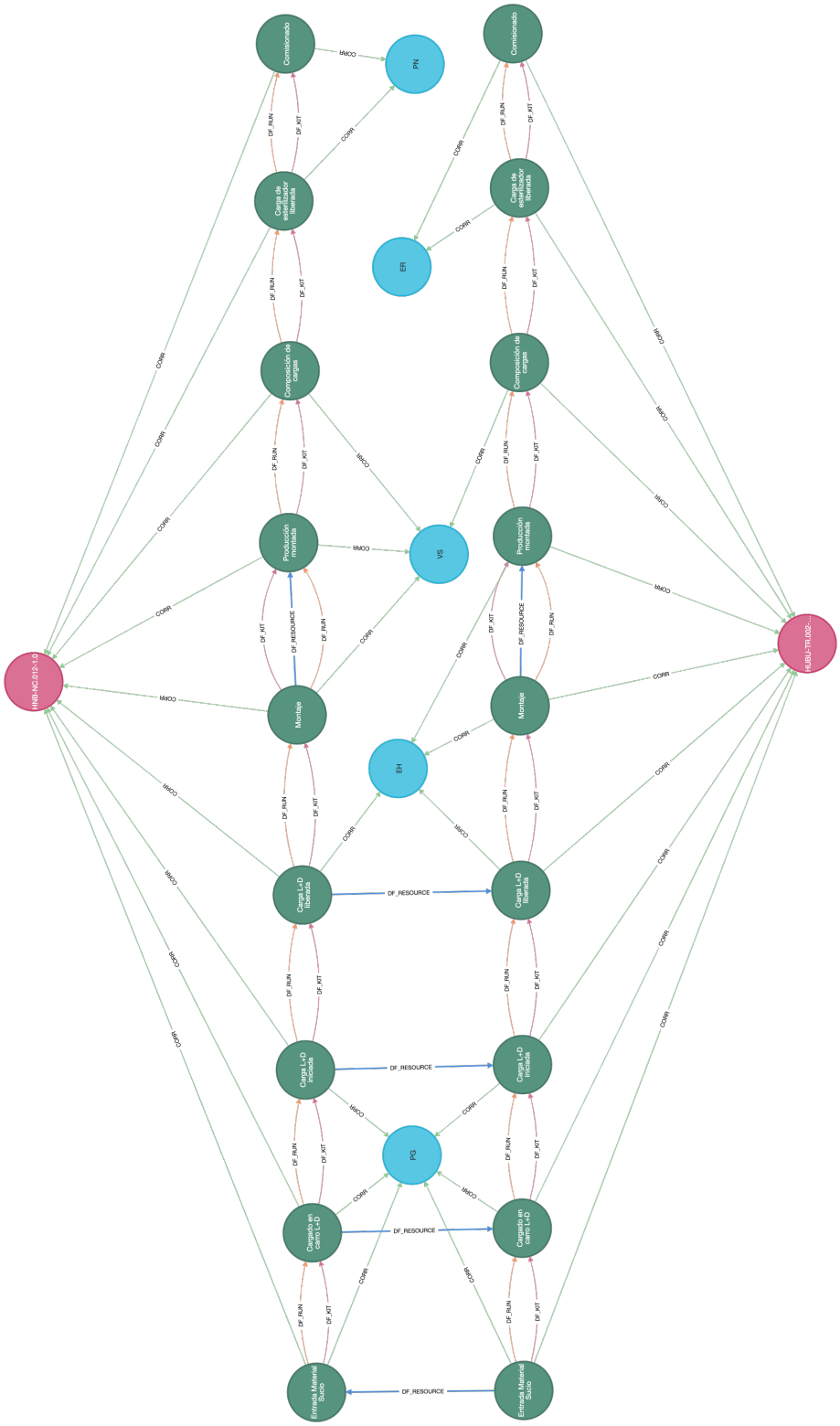

Figure 3.1: Event Knowledge Graph

the literature. For example, van der Aalst introduced the Piled Execution pattern in 2005, which closely aligns with the idea of batching [23]. Recognizing and understanding these patterns is crucial for the thesis, as it allows us to analyze how tasks are systematically organized and executed by human resources within the sterilization process.

Further, studies such as those by Wen et al., which focus on mining batch processing workflow models from event logs [32], and by Martin et al., which define batch processing and explore its identification in event logs [18], have significantly contributed to the understanding of batch processes.

Moreover, while Klijn, Mannhardt, and Fahland introduced a method for aggregating and analyzing tasks using EKGs to reveal patterns in resource behavior [15, 16]. This approach we use in our thesis from batch processing perspective and for revealing on the base of it resource working patterns. The paper initially defines Task Instances, further in groups it into clusters to create high-level events and subsequently analyse.

After applying the framework in this thesis analysis shows that the task instance analysis approach, while useful, does not fully capture the workflow of resources in the context of the sterilization process at Company C. The method focuses on tasks within a single case, but our study requires a broader perspective to understand how resources work across multiple cases throughout the day. Consequently, we propose a method that conceptualizes batching differently, focusing on resource behavior rather than solely on task instances.

Despite these differences, we do utilize some aspects of their methodology, particularly for defining aggregated relationships such as Node aggregation and Directly-Follows Aggregation.

Definition 10. ***(Node aggregation):*** *The first basic aggregation query* $Agg_{nodes}(a, X', \ell, \ell')$ *on an EKG* $G = (X, Y, \Lambda, \#)$ *proposed in [16] aggregates nodes* $X' \subseteq X$ *by property* a *into concept* ℓ *as follows:*

1. *query all values* $V = \{x.a \mid x \in X'\}$,
2. *for each value* $v \in V$ *add a new node* $x_v \in \ell$ *to* G *with label* ℓ *and set* $x_v.id = v$, $x_v.type = a$,
3. *for each* $x \in X'$, *add new relationship* $y \in \ell'$ *with label* ℓ' *from* x *to* x_v, $\overrightarrow{y} = (x, x_v)$.

Definition 11. ***(Directly-Follows Aggregation:)*** *The query* $Agg_{df}(t, \ell, \ell')$ *aggregates (or lifts) directly-follows (df) relationships between* `Event` *nodes for a particular entity type* t *to nodes in* ℓ' *along the* ℓ' *relationships as follows:*

1. *For any two nodes x, x' in ℓ, query the set $df^t_{x,x'}$ of all df-edges $(e,e')^t \in df$ where events $e, e' \in$ `Event` are related to x, x' via $y, y' \in \ell'$, $\overrightarrow{y} = (x,e)$, $\overrightarrow{y'} = (x',e')$.*

2. *If $df^t_{x,x'} \neq \emptyset$, create a new df-relationship $y^* \in df$, $\overrightarrow{y^*} = (x,x')$, set $y^*.type = t$ and set $y^*.count = |df^t_{x,x'}|$.*

The variant $\texttt{Agg}^{\neq}_{df}(\texttt{t},\ell, \ell')$ of the above query that requires $x \neq x'$ was proposed in [15].

Our approach builds on these studies and their definitions by applying batching within Event Knowledge Graph and introducing time constraints making it a novel approach. This allows us to study how tasks are batched over time, providing a clearer view of resource allocation in the sterilization process.

3.4.2 Human Resource Working Patterns

Another key area of focus in resource-oriented process mining is the analysis of human behavior patterns. Goel et al. discuss various patterns in the work. Two patterns highlighted in this paper are particularly relevant to our research: collaboration and utilization [11]. The collaboration pattern involves analyzing frequent sequence patterns to identify tasks often performed collaboratively by teams. Additionally, the utilization pattern, which focuses on workload-based task assignment, is central to the thesis resource analysis.

Importantly, these patterns might be additionally influenced by working shifts and weekdays, suggesting that resource behavior could vary significantly depending on the time of week and shift schedules [33, 17].

In the thesis study, we analyse collaboration pattern to examine how resources work together, identifying frequent sequences that indicate collaborative efforts withing shifts and weekdays. Along with that we study utilization pattern to understand how workload influences task assignment and whether processing efficiency is dependent on workload levels.

3.5 Analytical Techniques in Process and Data-Driven Research: Correlation and Sequence Analysis

This section covers correlation analysis, normalization, and frequent sequence analysis because we use these methods in the behavioral analysis in this

thesis. These techniques are important for this thesis research on resource behavior, and we refer to them to their key studies.

3.5.1 Normalization and Correlation Analysis

Normalization helps us compare different variables by placing them on a common scale. In this research, we use **Percentage of Total Normalization**, which scales each value as a proportion of the total sum of all values.

The formula for this normalization is:

$$x_i' = \frac{x_i}{\sum x}$$

where x_i is the original value, and $\sum x$ is the total sum of all values. In our approach, normalization applied buffer values and the high-level event log to ensure that the data is on a consistent scale. This step is important before performing correlation analysis, as if we can do correct comparisons between different time bins and activities.

Once the data is normalized, we apply **correlation analysis** to explore the relationships between various process attributes. Specifically, we investigate how the workload at specific stations relates to resource performance. This relationship is measured using the Pearson correlation coefficient:

$$r = \frac{\sum (X_i - \overline{X})(Y_i - \overline{Y})}{\sqrt{\sum (X_i - \overline{X})^2 \sum (Y_i - \overline{Y})^2}}$$

where X_i and Y_i are the data points, and $\overline{X}$ and $\overline{Y}$ are their means. Similarly, van der Aalst et al. [29] demonstrated the value of correlation analysis by uncovering previously unrecognized dependencies between different types of events in business process models. This technique is particularly useful for understanding how changes in one part of a process can affect outcomes in another. For instance in this thesis research, changing workload on processing of kits.

3.5.2 Frequent Sequence Analysis

Frequent sequence analysis identifies recurring patterns in datasets, often used in process mining to discover common workflows and predict events. Introduced by Agrawal and Srikant [3], and expanded by van der Aalst [2], this method reveals key process paths.

In our study, we use the FP-Growth algorithm [12], which mines frequent sequences of high-level batch activities among resources in a particular day of the week and a shift. We formally define frequent sequence and support:

Definition 12. ***(Frequent Sequence)****: A sequence* $s = \langle a_1, a_2, \ldots, a_k \rangle$ *is frequent if it meets a predefined support threshold.*

Definition 13. ***(Support)****: The support of* s *is the fraction of sequences in the dataset containing* s*.*

Using FP-Growth, we identify common high-level batch activity sequences in the high-level event log.

Chapter 4

Problem Exposition

In the previous chapter 2, we described the sterilization process and its business implications within Company C, and also provided a theoretical scientific base in the Chapter 3. Building on this foundation, we define the specific research questions for this thesis in Section 4.1. We then aim to identify working behavior and patterns in employee work using the research methodology described in Section 4.2.

Our main objective is to develop a framework that aggregates operational data into high-level features based on the timeline of a resource. This framework aims to capture user actions, providing a detailed perspective on resource activities over time. These high-level features, centered on specific employee behaviors and patterns, are important for constructing a digital twin designed to provide predictions and improve operational efficiency.

4.1 Detailed Research Questions

Given the primary goal, we define the following research question and associated subquestions:

Main research question: What are the identifiable patterns in employee work habits and task prioritization within Company C's sterilization process, and how can these patterns define the features provided for a digital twin that enhances operational efficiency and compliance?
Research subquestions:

1. Can the task instances implementation methodology from Klijn, Mannhardt, and Fahland work [15, 16] be effectively applied to present batching processing in the sterilization process and help in identifying and analyzing the working patterns of resources within Company C?

2. What additional batching detection techniques can be used to aggregate resource workflows, aiding in the identification and analysis of resource behavior in the sterilization process?

3. Are current batching techniques adequate for capturing the complex work behaviors of resources, or is there a need for more advanced aggregation methods to accurately identify and analyze work patterns?

4. How do employees prioritize tasks within the sterilization process, what factors influence their prioritization decisions?

5. Identify working patterns that could serve as indicators for features that explain work for further modeling in a digital twin?

4.2 Detailed Method

In this section we discuss the reseach mothodoly that was used in this thesis research.

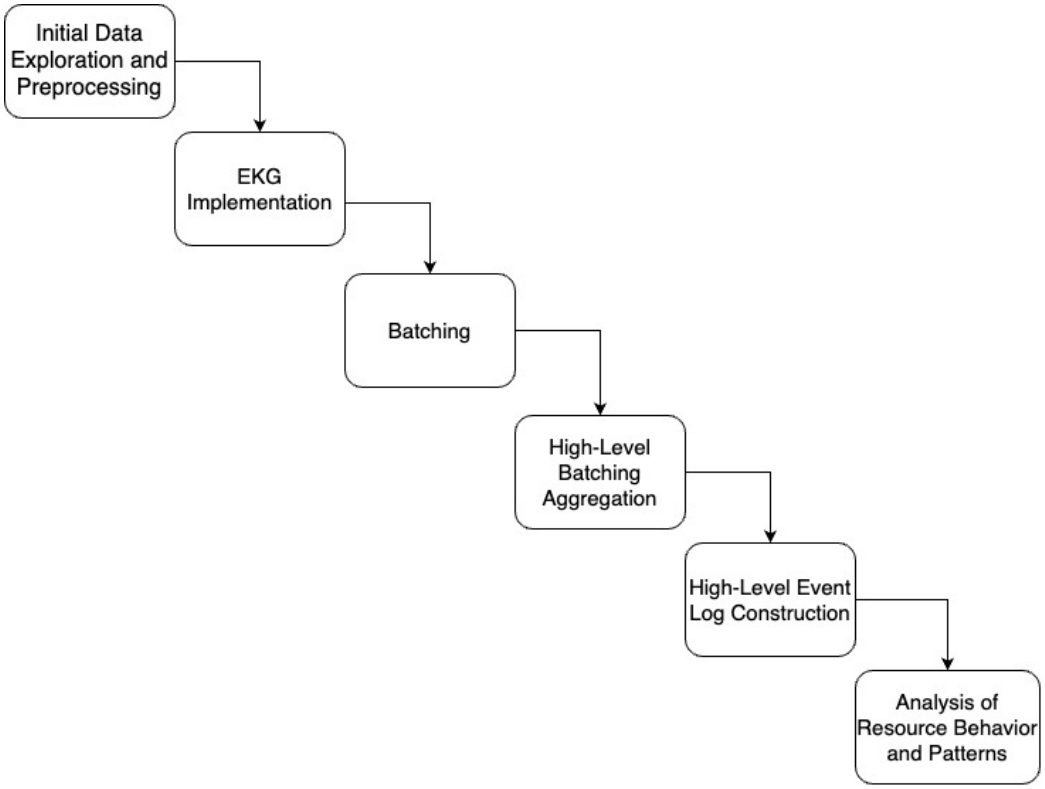

Figure 4.1: Thesis Research Methodology Outline

We provide an outline of this methodology in Figure 4.1 to generalize the approach into major steps. This figure captures the key phases of our research, each of which contributes to the overall analysis.

We employed the following methodology to understand the behavior and working patterns of resources in the sterilization process.

1. **Data Understanding and Business Understanding** The first step involved gaining a comprehensive understanding of both the data and the business processes involved of the company C.

   (a) **Review Documentation:** We examined existing documentation related to Company C's sterilization process to gather contextual knowledge and identify key variables and metrics. This review provided insights into the operational procedures. Additionally, we communicated with Company C to understand the business problem and their perspective on the goal for this thesis.

   (b) **Exploratory Data Analysis (EDA):** We conducted an exploratory data analysis to understand the structure, content, and quality of the available data. During EDA, we identified duplicated data and kits that disappears from processing after a particular sterilization stage provided in the Section 2.3.

2. **Data Cleaning and Preparation** In the data cleaning and preparation step, the focus was on ensuring the data was clean, and data issues in the Section 2.3.2 are resolved. This included removing duplicated data in the Section 5.1.1. Additionally we introduced `KitId` attribute that helps to resolve issue with kits that are presented only on particular steps.

3. **Graph Implementation, Statistics and Analysis** Using the cleaned data, we proceeded with graph implementation and resource behavior analysis in the Section 5.1.

   (a) **Running Graph Implementation:** We ran graph the provided by the AUTO-TWIN project with adaptations. This step allowed us to visualize the process and understand the interactions and flow within the sterilization process.

   (b) **Graph Implementation Statistics:** We provided implementation statistics regarding the graph nodes, entities and edge in the Section 5.1.2

(c) **Extended Resource Analysis Utilizing EKG:** Using insights from the graph analysis, we examined resource shifts and amount of people working on weekends and weekdays in the Section 5.2.

(d) **Performing Task Instance Analysis:** We tried to capture detected batching behaviour by using Task Instance approach introduced by Klijn, Mannhardt, and Fahland [15, 16] and reveal patterns in resource behavior in the Section 5.3.

4. **Implementation and Verification of Batching:** Due to insufficient results of implemented Task Instance Analysis, we proposed and implemented two batch modeling techniques in EKG in the Section 6.1: batching over resource in the Section 6.2 and batching over process activity 6.3, based on the earlier identified batching pattern. This involved modeling the batches within the sterilization process to gain detailed insights into how resources were utilized and managed.

   (a) **Batching Techniques Analysis:** We examined the two batching modelings techniques used in the sterilization process to understand their characteristics and implications. Batching over resource detects batches handled by a single user, focusing on how individual employees manage multiple kits. On the other hand, batching over station identifies batches processed with the same process activity, which can involve multiple users working simultaneously on a particular process activity. We implemented these batching models, analyzed their performance, compared the results, and selected the best approach, which was batching over station.

   (b) **Summarizing Resource Behavior Using Batching Techniques:** Batching over activity technique helped to explain and summarize resource behavior differently than on the level of individual events. However, we discovered complex workflows where there are presented overlapping batches that occur simultaneously, with resources alternating between multiple batches executed in parallel. The presence of overlapping batches motivated to propose a high-level batch aggregation method to capture these behaviors.

5. **Implementation and Verification of High-Level Batching:** After the initial batching, we applied a high-level batching technique to further aggregate the data presented in the Chapter 7. The high-level aggregation considered regular, iterative, and complex iterative resource

batch processing behavior, providing a more comprehensive view of resource working patterns by combining multiple BatchInstances into high-level batches.

Our requirements for high-level batching included eliminating iterative behaviors, capturing complex interactions, highlighting unique resource contributions. By implementing and verifying high-level batching, we aimed to gain a deeper insight into the resource behavior and working patterns, ultimately contributing to the thesis goal.

6. **Forming High-Level Event log:** Building on the high-level batching implementation, we constructed high-level event log. High-level event log is a log of ordered by time high-level batches for resources. High-level Event Log construction details are presented in the Section 8.1

7. **Analysis of Resource Working Behavior and Working patterns:** Utilizing the high-level event log, we conducted an in-depth analysis to understand resource working behavior and identify common working patterns. This analysis was essential for identifying patterns in employee work and task prioritization, which are key to developing an effective digital twin. The steps involved were:

   (a) **Finding Resource Working Behavior Based on High-Level Batching:** We analyzed resource working behavior within the sterilization process by examining high-level batch data. We performed correlation analysis between accumulated kits at stations formalized as workload and the corresponding actions taken by resources on the high-level batching aggregation in the Section 8.2.

   (b) **Identification of Common Working Patterns:** We identified common working patterns among resource in the Section 8.3 to provide a comprehensive understanding of the sterilization process. We used frequent sequence mining techniques to identify common patterns in high-level batch sequences, focusing on day-of-the-week and shift-specific behaviors. This analysis revealed typical working patterns and variations in resource utilization.

   (c) **Interpretation of Results:** We synthesized the findings from all analyses to draw comprehensive conclusions about resource behavior in the sterilization process. Additionally, we highlighted key insights that can inform future process improvements and digital twin implementation.

The code presented in the GitHub repository reflects the implementation of this methodology. While the actual data and Event Knowledge Graph used in this study are not included due to confidentiality agreements.

GitHub Repository: https://github.com/Liliia-Aliakberova/ThesisResearchHighLevelBatchAggregation

Chapter 5

From EKG Implementation to Extended Resource Exploration and Task Instances Analysis

In Chapter 3, we discussed that combining case and actor analysis is highly effective when using EKGs. Given the complexity of the sterilization process and the project's aim to accurately capture resource behavior and process flows, an EKG is essential for achieving these objectives. The AUTO-TWIN project initially provided a basic EKG, constructed using the method of Object-Centric Event Data Models [27]. This initial EKG included key entities such as resource, kits, runs, stations, sensors, production and activity.

However, we identified that further refinements were necessary in the EKG. Specifically, we needed to address the issue of kits disappearing after certain sterilization steps, an anomaly initially observed when analyzing the original dataset based on the Codigo attribute presented in the Section 2.3. For this issue we propose introduction of KitID attribute that will be incorporated into Kit nodes in the Section 5.1.1

Following the previous discussion in Section 5.1.2, we present implementation statistics along with resource exploration of Company C in the Section 5.2. In this section, we extend our resource exploration using the EKG for querying to understand shift organisation, resource number allocation during weekends and weekdays.

Finally, in the end of this chapter in the Section 5.3, we present the implementation of the Task Instance framework over the Company C EKG where we try to reveal if the approach applicable for presenting batching processing in the sterilization process.

5.1 EKG Implementation

Before the actual implementation of the Event Knowledge Graph (EKG) for Company C, significant preprocessing was undertaken to ensure the process data was represented in a correct way. This section details the preprocessing steps in the Subsection 5.1.1 and presents the subsequent implementation statistics of the EKG in the Subsection 5.1.2.

5.1.1 Preprocessing for EKG Implementation

For the construction of the Event Knowledge Graph (EKG), several critical steps were required to ensure the accuracy of the data and the comprehensive representation of the process.

Data Cleaning: To address issues with duplicate records discussed in Section 2.3.2, we implemented the duplicates cleaning. This involved identifying and removing duplicate entries from the dataset before loading the data into the Event Knowledge Graph (EKG). Removing these duplicates was important to ensure accurate process representation and unique tracking of each kit through the sterilization process.

Creation of KitID Attribute: We introduced the KitID attribute as for the Kit entity nodes within the EKG. This attribute, created by combining the "Código" and "N/S" attributes from the dataset, uniquely identifies each kit by merging its code and sequence number. Establishing the KitID was required for addressing the issue of kits disappearing after certain process stages, as noted in Section 2.3.2. The KitID enabled us to track kits through various stages and link separate activities that previously appeared isolated.

These steps in data cleaning and KitID creation were required in preparing the Company C dataset for EKG construction. In the next section we present implementation statistics.

5.1.2 EKG Implementation Statistics

After addressing the necessary data preprocessing and modifications as discussed in Section 5.1.1, the provided EKG for Company C was ran locally. The graph was built following the methodology delineated by Swevels et al [27] and incorporates the enhanced data elements to accurately represent the process interactions. The Table 5.1 presents the implementation statistics.

This table provides a detailed overview of the EKG components such as event nodes, resource nodes, and edges. The details provided in the table are

Table 5.1: EKG Implementation Statistics for Company C

EKG Component	Count
Total event nodes	219,019
Unique resource nodes	2,123
Kit nodes	2,123
Run nodes	12,121
Activity nodes	9
Total entity nodes (including Kit, Run, Resource, Activity, Stations, Production, Sensor, Connection)	74,397
Correlation edges (Kit and Run to events)	218,992
Correlation edges (Resource to events)	219,019
Directly follows edges (Kits)	216,896
Directly follows edges (Resources)	218,999
Directly follows edges (Runs)	78,748

mentioned specifically because they are important for this thesis and further analysis and implementations.

5.2 Extended Resource Exploration Utilizing EKG

Following the setup of the Event Knowledge Graph (EKG) for Company C, this chapter uses the EKG to conduct an extended initial exploration on resource utilization.

By leveraging the EKG, we can query specific data about resource behaviors, including how resources are scheduled across different shifts and days. Insights from literature [33, 17, 11] highlight the importance of such scheduling in influencing work patterns.

Through this exploration, we seek for initial insights from our event data in the EKG into shifts and weekday impact.

Resource Allocation Analysis Resource allocation during weekends typically involves two resources, aligning with standard operational protocols. However, a notable exception occurred on March 26, 2022, a Saturday, when three resources were deployed, possibly in response to an increased workload or specific operational demands. The number of resources during weekdays can vary significantly, from two to fifteen per day. On average, thirteen resources are active on weekdays, reflecting the dynamic nature of operational

demands throughout the week. For this, we utilized a Cypher query to aggregate the number of unique resources working across different weekdays. Importantly, these findings can inform the development of a digital twin, particularly for weekend operations, by providing a predictive baseline for the number of personnel likely to be working, which enhances the realism and accuracy of the simulation model.

Shift Organization In our analysis of resource shifts, we utilized Cypher queries to identify the earliest and latest event timestamps, allowing to determine the start and end times of each resource's working day.

To visually represent this information, we present two figures that display the distribution of these event timestamps.

The first figure, Figure 5.1, shows the distribution of the earliest event timestamps of resources. This graph illustrates when resources typically begin their work, highlighting the start times across different dates. The data points are spread out, showing variability in when resources start their day.

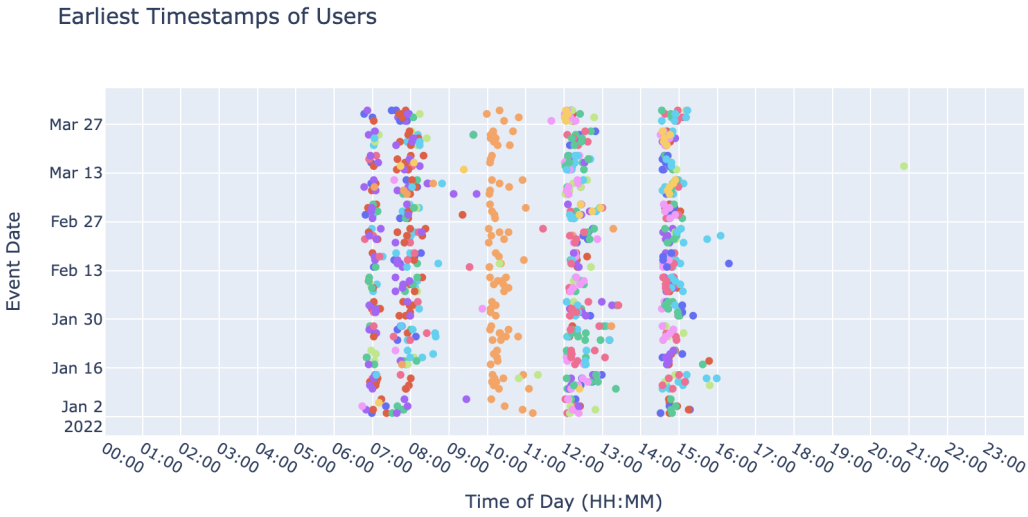

Figure 5.1: Distribution of Earliest Event Timestamps of Resources

The second figure, Figure 5.2, presents the distribution of the latest event timestamps of resources. This visualization demonstrates when resources typically finish their work. Similar to the start times, the end times are spread across various hours.

What we can observe is that resources are generally organized into 2 to 4 groups, depending on the weekday, with minor outliers. We will refer to them

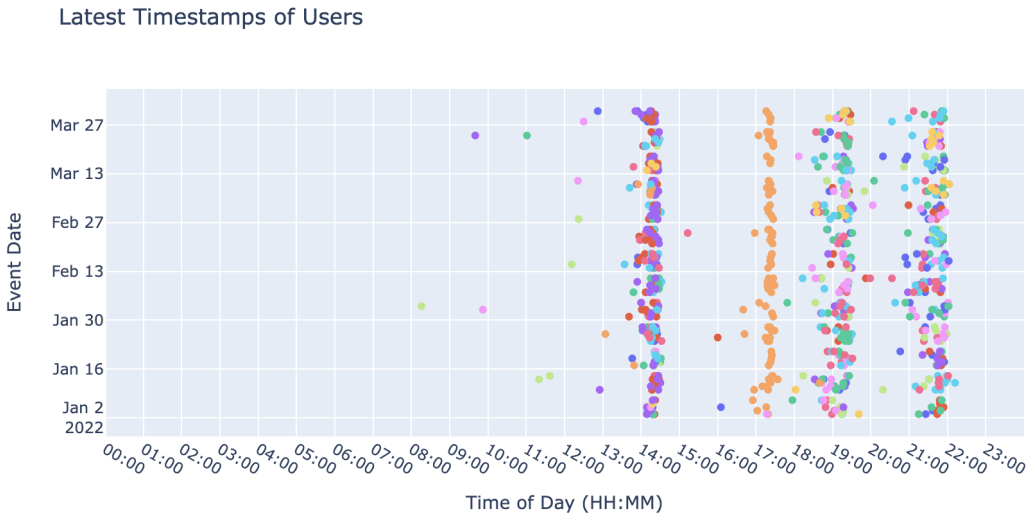

Figure 5.2: Distribution of Latest Event Timestamps of Resources

as shifts. These shifts are likely designed to ensure continuous coverage and maintain optimal productivity throughout the day, with an average interval of 2 to 2.5 hours between each shift. In our analysis, we observe that on weekends, there are primarily 2 shifts, while on weekdays, the number of shifts ranges between 2 and 4. Based on the Figure 5.1 and Figure 5.2 these shifts are structured as follows:

- **00:00 to 09:30:** The early morning shift typically handles initial operations, setting the pace for the day's activities.
- **09:30 to 11:30:** The mid-morning shift often experiences an increase in activity as it overlaps with peak operational hours.
- **11:30 to 14:00:** The late morning to early afternoon shift continues managing the workload, ensuring continuity and addressing any backlogs from earlier in the day.
- **14:00 to 23:59:** The afternoon to late evening shift covers the remainder of the day, ensuring that operations are sustained until the end of the business day.

This analysis of shifts, based on actual start and end timestamps of event nodes correlated to resources, helps us understand how resources are allocated across different parts of the day. The information gathered is also be valuable for development of Company C digital twin and the further in depth high-level resource working patterns analysis.

5.3 Applying Task Instance Framework over Company C's EKG

We further proceed to the Task Instances Analysis. In Chapter 3.4.2, we discussed the intention to apply the Task Instance Framework to aggregate EKG to analyse batching processing from resource perspective. This approach is based on the methodology introduced by Klijn, Mannhardt, and Fahland in their study on aggregating event knowledge graphs for task analysis [15, 16]. We aimed to apply this framework on Company C's sterilization process, using the EKG to gain insights into batching activities and resource behavior within the process.

5.3.1 Limitations of the Approach

We encountered challenges in directly applying the Task Instance Framework to the EKG of Company C's sterilization process. The original framework, while effective in certain contexts, did not fully align with the complexities and unique characteristics of the data we were working with. To address these challenges, we adapted the framework for EKG of Company C. However, we can notice significant limitation for our data.

The primary limitation of applying the Task Instance Framework to the sterilization process at Company C lies in its inherent focus on task instances within a single process case, particularly a kit case. This framework is designed to emphasize the process flow related to individual kits, which can be useful in understanding how resources interact with specific cases. However, this approach tends to narrow the analysis to the kit process itself, rather than providing a view of the broader working patterns and behaviors of resources throughout the day.

Our implementation revealed that applying the Task Instance Framework to Company C's process did not provide the expected insights. A key issue was the limited number of process stages in the sterilization process, which led to the aggregation of several stages into a single high-level event. This single event, treated the same as normal events in the graph, did not capture the detailed behaviors of resources. This limitation was particularly evident for resources with lower levels of activity, like "DF" and "MGP", whose contributions were not sufficiently represented, making it difficult to discern their specific behaviors and interactions within the process.

These challenges underscore a critical need for an alternative approach that shifts the focus from process cases to a more comprehensive analysis of resource behavior across various tasks and activities throughout the workday.

5.3.2 Need for Batch Detection Revision

Given the limitations of the Task Instance Framework, our analysis pointed towards the importance of identifying and understanding batching behaviors within the resource workflows. Batching, where resources perform several similar tasks in sequence. In the next chapter, we propose two batching techniques that can fit better for the data.

Chapter 6

Batching

The goal of this thesis is to understand the working patterns and behavior of resources within Company C's sterilization process. The Event Knowledge Graph and Implemented Task Instances Analysis Framework do not provide sufficient insight into these patterns and behaviors. Since, we identified in the previous Section 5.3 that resources often work in batches. This is a challenge, thus we believe that representing and summarizing resource work through batching is an improved approach compared to the existing representation.

To address this issue, we need to establish batching modeling techniques. In this thesis, we propose two techniques: batching over resource and batching over station. These techniques will help us summarize the resource workflow within the sterilization process. Batching in the thesis context refers to the grouping of event nodes that are correlated to certain entity nodes, such as the same resource or activity, and occur within a defined time window.

We begin by examining the general concept of batching in Company C's EKG and proposing different modeling techniques in Section 6.1. We first formalize the Batching over Resource technique in the Section 6.2. Following this, in Section 6.3, we formalize the second technique, Batching over Activity. Finally, in Section 6.3.3 we discuss and compare these approaches, and identify the most suitable one for Comapny C.

6.1 Batching in the Event Knowledge Graph: Proposing Options for Company C

In order to develop a correct batching detection approach for Company C, it is required to address the limitations identified in the methodology presented in Section 5.3 and establish a method for conceptualizing batching.

The task instance analysis based on the work by Klijn, E.L., Mannhardt,

F., and Fahland, D. [16] specialises on the task instances creation for resources within one particular process case, specifically for the Company C this is a kit case. This means the analysis focuses more on the kit process flow rather than capturing the real behavior (workflow) of the resources within a day. Consequently, it fails to reveal the patterns where resources perform events in groups. In contrast, we need an approach that focuses on resource behavior, which is the main difference from Eva Kleijn's work.

To overcome these limitations, we need to discuss what conceptualize a batch processing. Batching is a process in which similar tasks or items are grouped together and processed as a single unit [31]. From the Company C' EKG perspective the idea of a batch involves presence of Event nodes with similar node attributes and correlated to the common entity.

For batching in the Company C' EKG we can consider Resource, Activity and Kit Entities because they are main components of the process that interact frequently and impact the workflow. However, it is important to evaluate each entity:

- Resource: We can conceptualize batching by examining how individual employees handle same type of work specifically multiple kits over time on the same activity. This approach can provide insights into resource utilization, their generalized workflow and task prioritization.
- Activity: By considering an activity as an entity, we can identify batches of events performed at the same process stage. This method allows us to detect multiple events occurring simultaneously or in close succession at the same process stage (station) regardless the resource.
- Kit: Considering kits alone does not constitute batching since each kit is processed individually.

Thus, we consider defining batching using resources and activity entities. In this context, we define a batch as a group of events that occur within a specific time window, based on the average arrival time of kits at the initial station "Entrada de Material Sucio" where kits start the process. We analyzed the arrival time and detected that the average arrival time is 5 minutes. A batch considers a singular event and multiple events.

Option 1: Batching over Resource

This option groups events by the resource performing the activities, based on a sequence of event nodes along the directly-follows edges of the Resource (DF_RESOURCE). The logic for batching in this approach is defined by three primary criteria:

1. **Execution by the Same Resource:** Events must be executed by the same resource to be considered part of the same batch. In the EKG concept, this means that we group into one batch event nodes that have a correlation relation to the same resource node. This ensures that the grouping reflects the specific tasks being performed by a single resource.

2. **Performance of the Same Activity:** Events must involve the same activity to be included in the same batch. This criterion ensures that the events are related to the same type of work or operation. In the EKG concept, all event nodes joined into a batch must have an equivalent activity attribute.

3. **Temporal Proximity:** Grouped events must occur in consecutive order within a 5-minute window based on their event timestamps. This ensures that the events are temporally close to each other, reflecting a continuous workflow by the same resource. In the EKG concept, the difference between the timestamp attributes of subsequent events along the directly-follows edges of the Resource should be within 5 minutes.

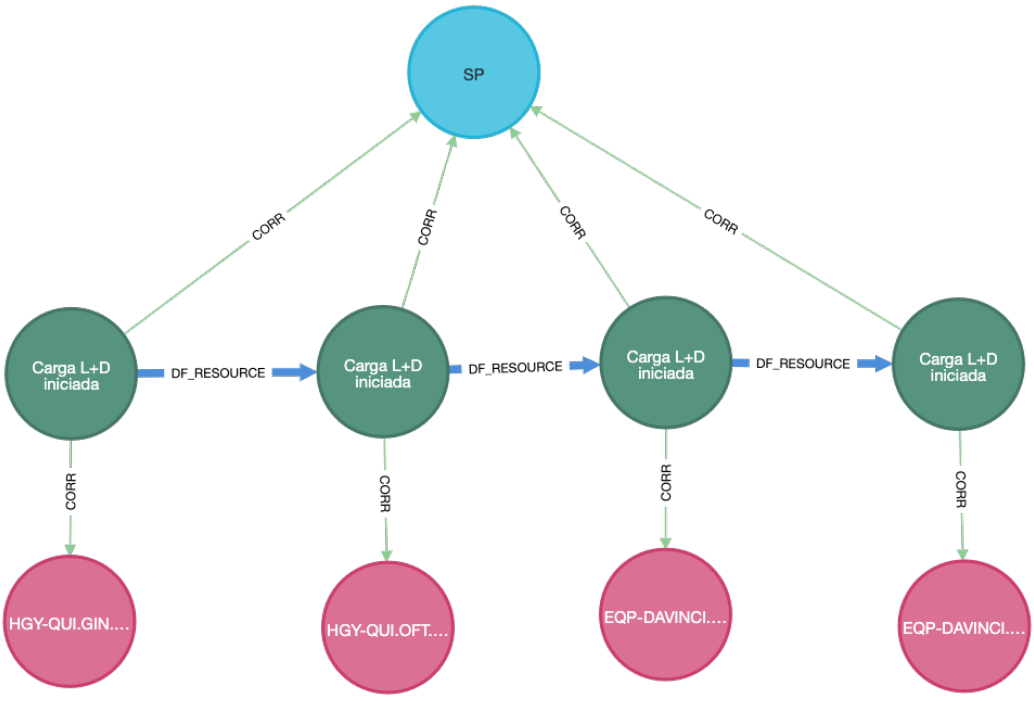

Figure 6.1: Example of Batching over Resource in the Event Knowledge Graph

Provided Figure 6.1 figure demonstrates the Batching by Resource option within the EKG. It showcases a series of green event nodes, all labeled "Carga L+D iniciada," which indicates that each of these events involves the same activity type. These nodes are connected through directly-follow relationships 'DF_RESOURCE', reflecting the sequence of operations performed by the same resource 'SP'. For the resource we have a blue node and pink nodes for kits are processed.

This visualization captures a specific instance from the date 12.01.2022 at 14:36, illustrating continuous operational flow of tasks that are temporally close within a 5-minute window. Each node's transition from one to the next without significant time lapses fulfills the criteria for batching based on execution by the same resource, performance of the same activity, and temporal proximity.

This setup shows how we can group event nodes by resource in the EKG for Batching over Resource approach.

Option 2: Batching over Activity

Another option is to group events based on the Activity Entity. The criteria for batching include:

1. **Equivalent Activity:** Events must involve the same activity to be included in the same batch. This ensures that the grouping is meaningful and reflects the specific tasks being performed at the station. From the EKG perspective, all event nodes joined into a batch must have an equivalent activity attribute, since we additionally store the Name attributes of Activity Nodes as an attribute of Event nodes.

2. **Temporal Proximity:** Grouped events must be ordered by their timestamps attributes, with the time difference between consecutive events being less than 5 minutes. This criterion ensures that the events are temporally close to each other, reflecting a continuous workflow.

To illustrate Batching over Activity in the EKG we provide following Figure 6.2. This figure displays several green event nodes, each labeled "Cargado en carro L+D". These nodes are linked by observed edges to an Activity node with the same label, indicating that similar tasks are performed at this stage of the process.

The visualization shows these event nodes with timestamps from 10:48:00 to 10:49:00 on 2022-03-15, highlighting the parallel execution of the same activity by several resources. Each event node is connected to blue resource

nodes and pink kit nodes through correlation relationships, demonstrating the integration of resources and kits a within this operational process stage.

This arrangement demonstrates parallel task execution, serving as a clear example of how we can groups similar event nodes in the EKG.

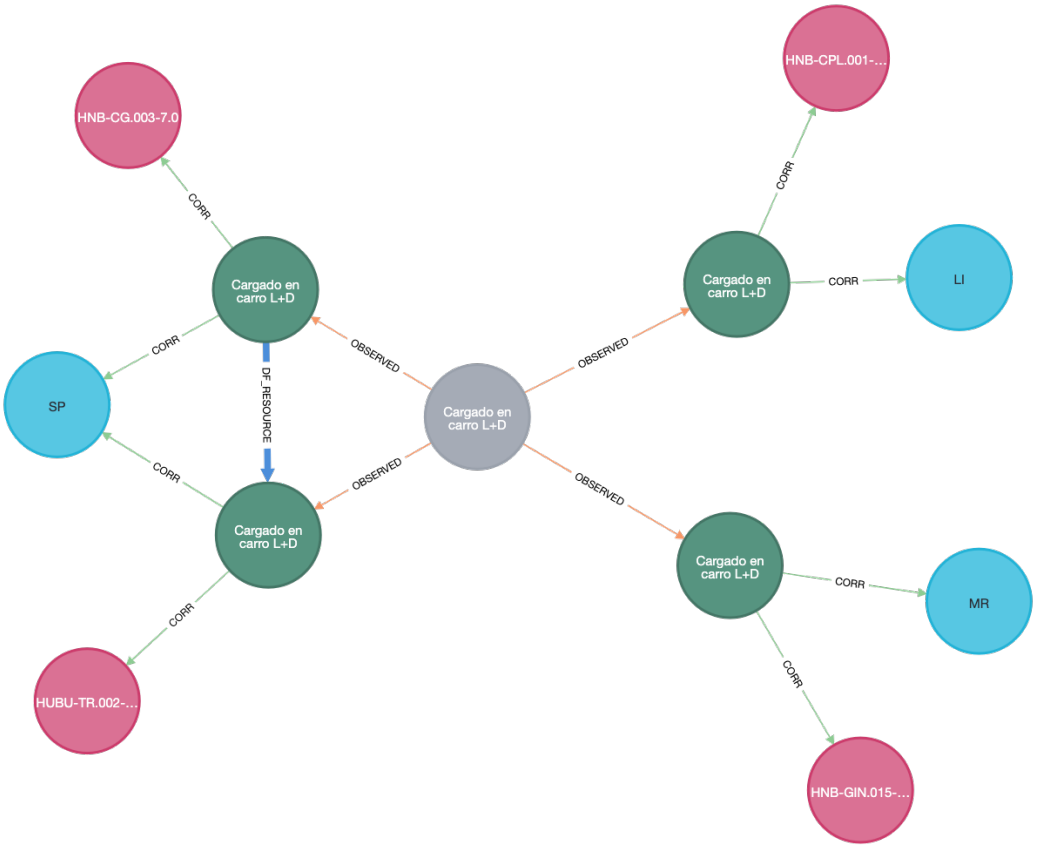

Figure 6.2: Example of Batching over Activity in the Event Knowledge Graph

In the following sections, we formalize each of the approaches, providing detailed implementation steps.

6.2 Batching over Resource

In the previous section, we introduced the batching over resource approach. The method focuses on groups of events based on the resources performing them, their temporal proximity, and activity. It is important to note that a batch can consist of a single event or multiple events satisfying the criteria above.

In the current section we formalized this method by presenting its mathematical definition and implementation steps. Additionally, we provide provide an example of batching on the Figure 6.3.

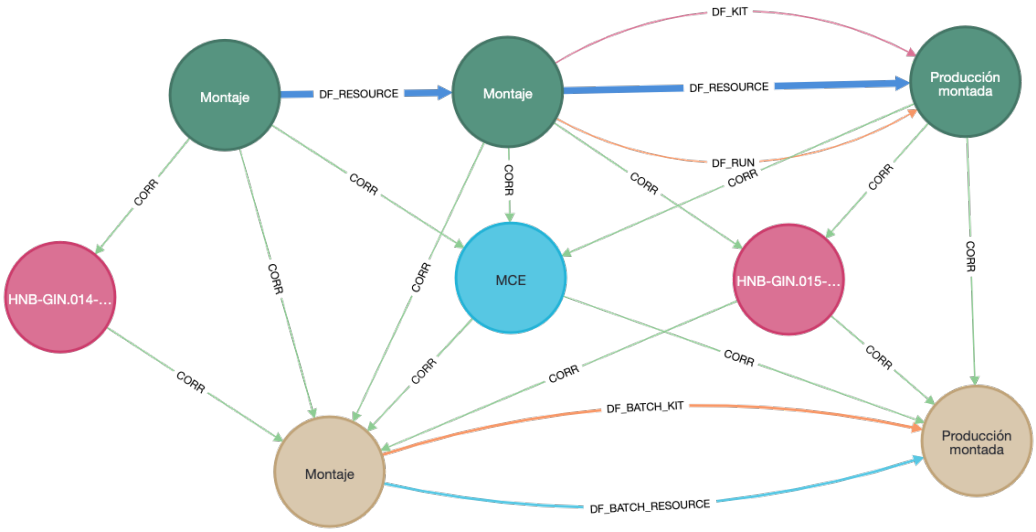

Figure 6.3: Example of Batching over Resource

In addition to the visual representation of the Batching over Resource approach we provide BatchInstance nodes attributes description in the Table 6.3

Based on the Figure 6.3 and Table 6.3 we can observe that green nodes represent event nodes, pink nodes represent nodes of kits, the blue node represents the resource node, and the light brown nodes represent the BatchInstance nodes themselves. The event nodes are connected with DF_RESOURCE edges, highlighting the sequence of activities performed by the resource. Additionally, we see that batches are connected with DF_BATCH_RESOURCE and DF_BATCH_KIT edges, indicating the relationships between different batches performed by the same resource over kits. This example illustrates how events are grouped into batches and demonstrates the sequence and relationships between different activities and resources. Specifically, we see how the resource "MCE" performs activities such as "Montaje" and "Producción montada" within the 5-minute window, processing different kits and demonstrating the workflow continuity for this resource.

Table 6.1: BatchInstances Attributes for Batching over Resource Approach Visualization

	BatchInstance 1
Activity	Montaje
Batch_number	14
Earliest_timestamp	"2022-01-01T10:01:00Z"
Latest_timestamp	"2022-01-01T10:03:00Z"
Events_number	2
Kits	[HNB-GIN.014-33.0,HNB-GIN.015-9.0]
Kits_number	2
Runs	[HNB-GIN.014-33.0-0,HNB-GIN.015-9.0-0]
Resource_sys_id	MCE
	BatchInstance 2
Activity	Producción montada
Batch_number	15
Earliest_timestamp	"2022-01-01T10:05:00Z"
Latest_timestamp	"2022-01-01T10:05:00Z"
Events_number	1
Kits	[HNB-GIN.015-9.0]
Kits_number	1
Runs	[HNB-GIN.015-9.0-0]
Resource_sys_id	MCE

6.2.1 Mathematical Definition of Batching over Resource

Definition 14. ***(Batch over Resource)*** *The mathematical formula for defining batches over resource based on the criteria mentioned in the Section 6.1 is as follows:*

$$Batch_{resource}(e_i, e_j) = \begin{cases} 1 & \textit{if } resource(e_i) = resource(e_j) \\ & \textit{and } activity(e_i) = activity(e_j) \\ & \textit{and } |timestamp(e_i) - timestamp(e_j)| < 5 \textit{ minutes} \\ 0 & \textit{otherwise} \end{cases}$$

Where: - e_i and e_j are consecutive events. - resource(e) represents the resource performing the event. - activity(e) represents the activity of the event. - timestamp(e) is the time the event occurred.

This formula helps in determining whether two events belong to the same batch based on the defined criteria.

6.2.2 Batching over Resource: Implementation Steps

The batching over resource process involves several steps:

1. **Assigning Batch Attributes:** The primary issue we address is the need to associate each event node in the EKG to a specific batch. To solve this, we introduced a "batch" attribute for each event node, assigned numeric value that identifies a specific batch of event group. The assignation performed in the following order:

   (a) **Sequential Evaluation:** Event nodes are sorted and assessed based on their `timestamp` attribute to maintain chronological order, alongside the `system ID` of the correlated resource node, ensuring that each event is performed by the same person.

   (b) **Grouping Criteria:** Each event node is compared with the preceding one to determine if they share the same `activity` attributes within a 5-minute window, as marked by the `timestamp`.

   (c) **Batch Attribution:** Events that align under the grouping criteria are assigned the same batch number. If the criteria are not met, a new batch is started. This batch number is then recorded as the 'batch' attribute for each event node.

   This step insures classifying the event nodes by batch.

2. **Creating BatchInstance Nodes and Connecting to Event Nodes:** Once the batch attributes are assigned to event nodes, we proceed with creation of BatchInstance nodes, that present these batches in the EKG. Each BatchInstance node represents the events that share the same resource and activity within the 5-minute window. A BatchInstance node includes the following attributes:

   (a) `Activity` – represents the activity of all associated events of a batch.

   (b) `Batch_number` – numerical identifier of a batch, ensuring each batch is uniquely identifiable. This helps in tracking and referencing specific batches easily.

(c) `Earliest_timestamp` – a timestamp of the first event in the batch, marking the start of the batch. This indicates when the batch processing began.

(d) `Latest_timestamp` – a timestamp of the last event in the batch, marking the end of the batch. This indicates when the batch processing ended.

(e) `Events_number` – the number of events included in the batch, providing a quantitative measure of the batch size. This helps in understanding the workload and activity level within each batch.

(f) `Kits` – a list of kits included in the batch, detailing the specific items involved. This attribute provides insights into the materials or items processed together.

(g) `Kits_number` – the number of kits included in the batch, indicating the diversity of items processed. This helps in assessing the complexity and variety of the batch.

(h) `Runs` – a list of runs associated with kits in the batch, reflecting the operational flow. This attribute helps in understanding the sequence and flow of operations within the batch.

(i) `Resource_sys_id` – system name of the resource that performed all events in the batch. This helps in tracking the personnel involved in each batch.

Example of created BatchInstance nodes presented on the Figure 6.3 where the nodes colored in ligh-brown.

Further, BatchInstance nodes are connected to Event nodes with correlation relationships (CORR) based on the shared batch number attribute . This edge in EKG ensures that all events included in a batch are associated with the corresponding BatchInstance. Events within the same batch meet the criteria established in the earlier section 6.1. On the Figure 6.3 we can observe green edge CORR that represent established correlated relationship between green Event nodes and light-brown BatchInstance nodes.

3. **Establishing Relationships:** The final step involves creating the rest of correlation relationships (CORR) between BatchInstance nodes Resource, and Kit nodes and creation of directly-follow relationships between BatchInstance nodes:

   (a) **Correlated Relationships:** Each BatchInstance node is linked to the corresponding Resource and Kit nodes to accurately rep-

resent their interactions within a batch. Specifically, BatchInstance nodes are connected to Resource nodes based on the shared resource's system ID, and to Kit nodes by associating through shared kit identifier. This ensures that all resources and kits involved in the batch are properly represented. In the provided Figure 6.3, we can observe this type of relationship as a green edge named CORR between pink nodes representing Kit nodes, while the blue node representing the Resource node and light-brown BatchInstance nodes.

(b) **Directly-Follow Relationships:** This process includes creating two types of directly-follow relationships DF_BATCH_KIT and DF_BATCH_RESOURCE. These relationships are constructed following the methodology of directly-follows aggregation outlined by Klijn, Mannhardt, and Fahland [16] and in the Section 3.4.1. By establishing these directly-follow relationships, we create a detailed view of the workflow within the sterilization process of kits and resource on the batching aggregation level. Relationships DF_BATCH_KIT were extended with attributes such as `kitId` and `runId`. This modification helps distinguishing which edge is associated with each kit, particularly when multiple incoming and outgoing edges are present. On the Figure 6.3, these relationships for Resources are represented as blue edges named DF_BATCH_RESOURCE and for Kits DF_BATCH_Kit connecting two light-brown BatchInstance nodes, illustrating how batches are sequentially linked.

With this final step, we established the required relationships within the Event Knowledge Graph, allowing us to view a fully aggregated subgraph that illustrates batching over resource.

To sum up, the implementation steps of Batching over Resource detection methodology in this subsection provide an approach for grouping event nodes into BatchInstance nodes and enriching EKG. Initially, it groups event nodes into BatchInstance nodes. It then establishes relationships between the batches and their corresponding events, resources and kits and after within these batches. By linking these components, we create a subgraph within the EKG, enabling visualization of the batching processes, as demonstrated on Figure 6.3.

6.2.3 Batching over Resource: Implementation Results and Exploration

In this section, our goal is to assess the effectiveness of the Batch over Resource approach. We begin by reviewing implementation statistics and conducting an exploration. Following this, we will discuss the approach in detail. The implementation of this approach resulted in:

Table 6.2: Summary of Implementation Results for Batch over Resource Approach

Entity	**Count**
BatchInstance Nodes	53,646
Correlation Relationships	430,714
Directly-Follow Relationships for Kits	288,016
Directly-Follow Relationships for Resources	107,252

Small Batches Initially, we explored the distribution of events within batchInstances to understand the underlying patterns and frequency distributions.

We examined the distribution of events in BatchInstances. The scatter plot 6.4 displays the number of events within each batch instance on the x-axis and the frequency of these instances on the y-axis.

The distribution of BatchInstance nodes, characterized by a predominance of instances with only a few events, raises questions about the alignment of the current batching method with operational realities. Notably, the sharp decline in the occurrence of larger batches may indicate challenges in managing and processing these batches, potentially due to resource constraints or inefficiencies.

Upon further analysis, we identify see this a notable notable limitation of the current batching over resource approach. is its tendency to generate a substantial portion of batches. Approximately 58% of BatchInstances consist of only a single event.

The creation of single-event batches undermines the primary advantage of batching, which is to group multiple related events for more efficient processing. Thus, we note several important aspect of this limitation:

1. **Ineffective Data Aggregation:** The frequent occurrence of batches containing only a single event suggests that the current method of

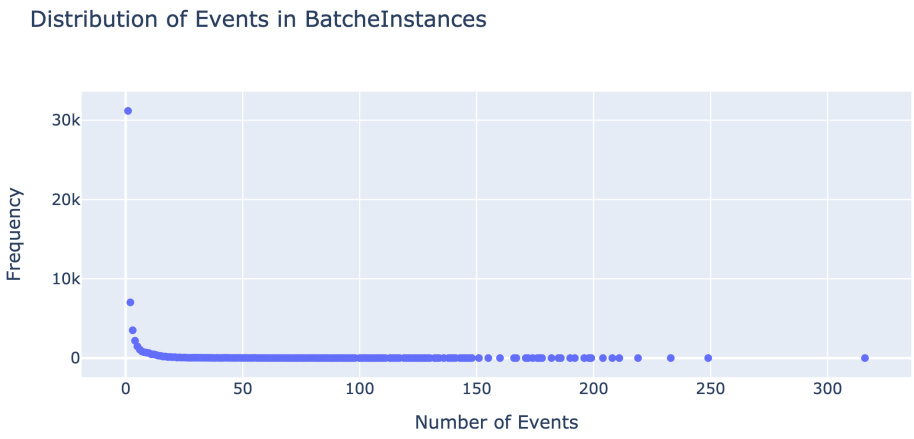

Figure 6.4: Distribution of Events in Batchinstances for Batching over Resource Approach

batching does not effectively aggregate data, leading to limited improvements in operational efficiency or data manageability.

2. **Inadequate Grouping Mechanism:** The existing method of forming batches does not appear to facilitate a meaningful or practical grouping of work, which might suggest a misalignment with actual operational practices.

3. **Reevaluation of Batch Formation Principles:** The observed patterns in batch sizes and related challenges may indicate a misunderstanding of how batching is typically implemented in practice. This implies that the current conceptualization of batching may not accurately reflect the diverse ways in which individuals organize and execute their work.

Resources Return to Previous Activities The current batching logic creates new batch nodes each time a resource moves to another activity, even if they return to the original activity shortly after. This approach results in the fragmentation of what could be considered a single, cohesive batch into multiple smaller batches, reducing the efficiency and clarity of resource working patterns. An example of the pattern presented on the Figure 6.5

The Figure 6.5 shows how SP resource performs a sequence of batches within a short time frame on March 23, 2022. Initially, at 14:45, SP is involved in a batch with the activity "Cargado en carro L+D", handling

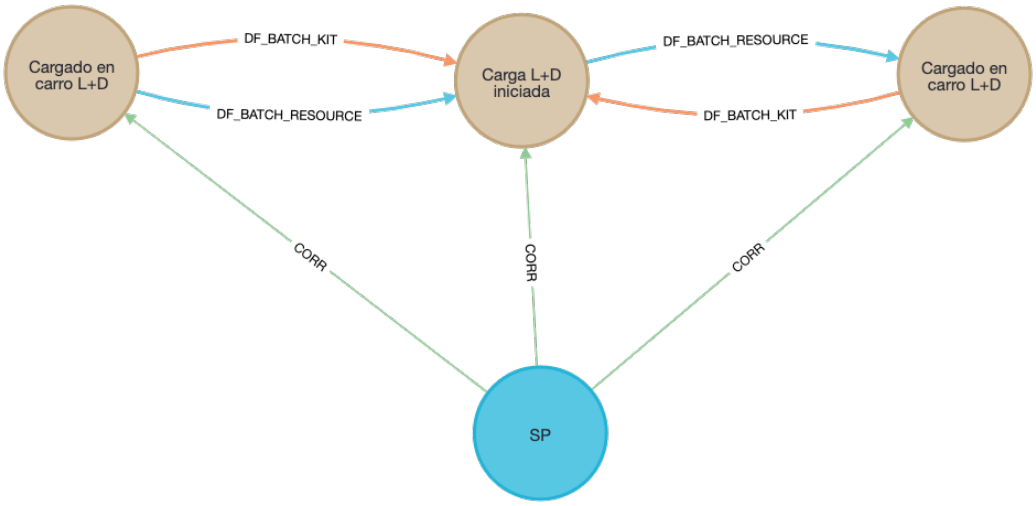

Figure 6.5: Example of Resources Return to Previous Activities in EKG

two kits. Almost immediately afterwards, at 14:46, SP moves to a different batch with the activity "Carga L+D iniciada", working on the same set of kits. The resource then performs at 14:46 a new batch with the activity of the initial batch "Cargado en carro L+D". This subgraph demonstrates resource's ability of returning to the same type of work switching very shortly on another type.

We analysed company data to verify presence of more cases where resources perform batching in one activity, proceed to another activity, and then return to the initial activity within a 5-minute time frame. For this we constructed a query that search for a pattern within the EKG,the query finds sequences of three BatchInstance nodes connected by DF_Batch_Resource edges. Each node in this sequence is correlated to the same resource node, and the entire sequence falls within a strict 5-minute time frame, reflecting the constraints on batch creation times.

Our query was designed to specifically identify sequences where the first and the last BatchInstance nodes in the pattern share the same activity attribute of BatchInstance nodes, indicating a return to the initial activity after an intermediate change. The middle node in the sequence is required to have a different activity attribute of a BatchInstance node, showcasing a temporary shift in task focus.

We present results on the Figure 6.6 that illustrates the most frequent sequences with a pattern.

In the visualization, it is evident that resources frequently oscillate between activities within short timeframes, leading to the creation of frag-

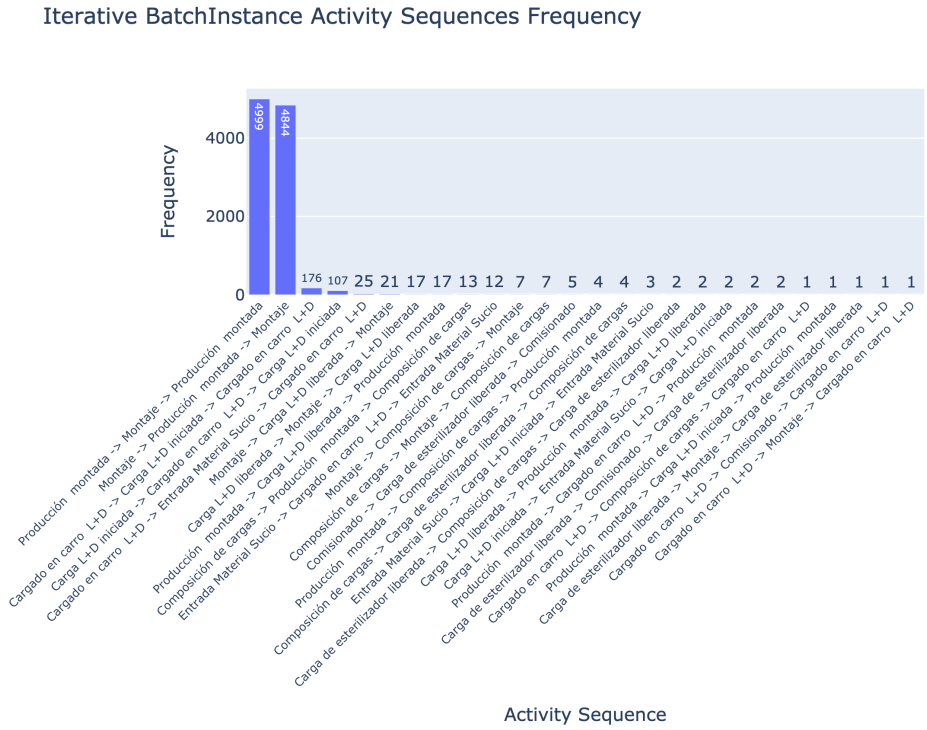

Figure 6.6: Iterative BatchInstance Activity Sequences Frequency

mented batches. The bar chart displays the frequency of various activity sequences, highlighting that certain sequences occur repeatedly within a short period. For instance, sequences such as "Montaje -> Producción montada -> Montaje" and "Producción montada -> Montaje -> Producción montada" appear with high frequency. This pattern indicates that resources often return to their initial activity after completing another, within a 5-minute window.

This not only complicates the tracking of resource activities but also undermines the potential benefits of batching by failing to capture the continuity of resource usage.

Concurrent Resource Utilization Another significant limitation of the current approach is its inability to accurately represent scenarios where multiple resources work together on the same activity within the same time window. An example of concurrent resource utilization presented on the Figure 6.7.

The Figure 6.7 presents a scenario where multiple resources, represented

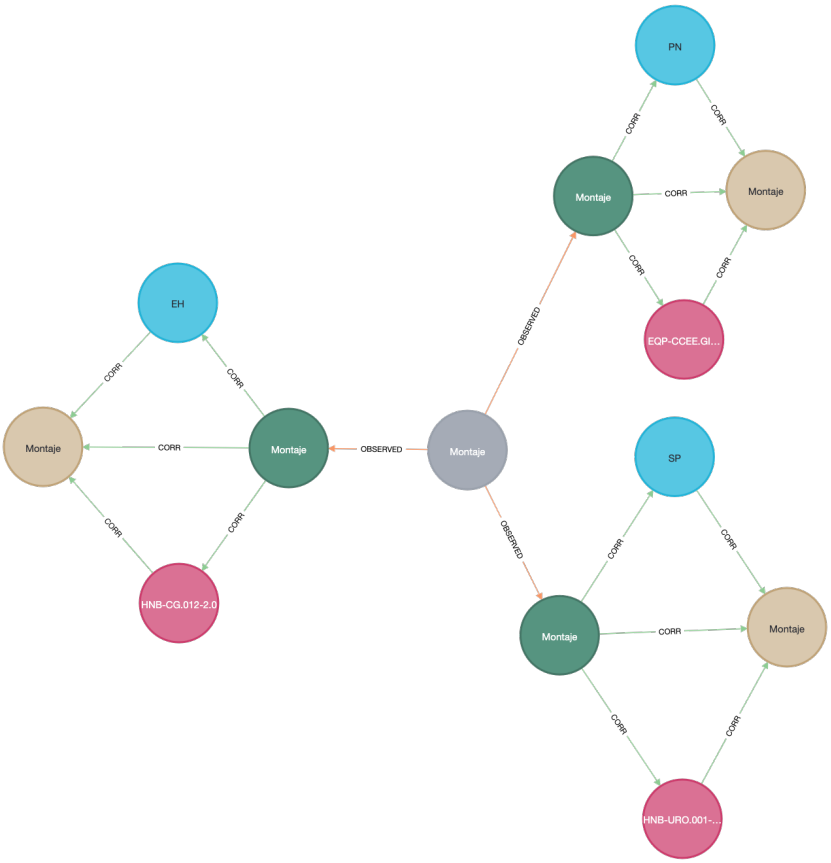

Figure 6.7: An example of concurrent resource utilization

by blue nodes (EH, PN, and SP), simultaneously working on the activity "Montaje," depicted by the gray node. Each resource is connected to a green event node, which in turn is linked to a light brown BatchInstance node and pink kit node, indicating that while all resources are involved in the same activity on 2022-03-31 at 14:59 and each operates within its own distinct batch.

Analysis shows that several resources often perform events on the same activity within a 5-minute timeframe. The existing batching logic fails to account for this collaborative aspect, instead creating separate batches for each resource even when they are effectively working together.

To address this, we utilized a query within the EKG to identify BatchInstance nodes that are connected to different Resource nodes yet share an equivalent activity attribute. The query specifically searched for BatchInstance nodes whose activity attributes matched and timestamp attributes occur within a 5-minute window of each other. The visualization on the

Figure 6.8 illustrates the results of the query.

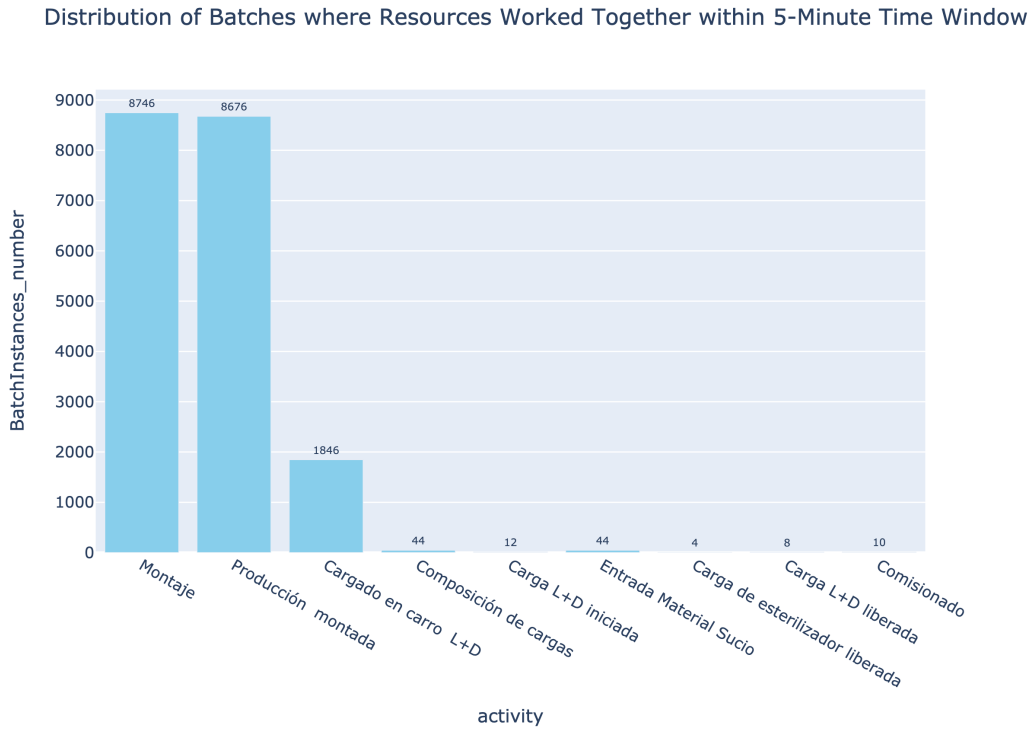

Figure 6.8: Distribution of Batches where Resources Worked Together within 5-Minute Time Window

In this bar chart 6.8, the x-axis represents different activities, and the y-axis shows the number of batchInstances where multiple resources worked together within a 5-minute time window. The chart indicates that activities such as "Montaje" and "Producción montada" have a significantly higher number of concurrent resource batches, with counts of 8746 and 8676, respectively. This suggests that these activities are more likely to involve collaborative efforts among multiple resources.

Other activities, such as "Cargado en carro L+D" and "Composición de cargas," also show instances of concurrent resource utilization, though to a lesser extent. The presence of multiple resources working simultaneously on the same activity suggests that the batchInstances created under the current logic may not represent the actual operational scenarios accurately.

This limitation is important because it affects the understanding of resource interdependencies and the overall workflow. By not accounting for repetitive resource patterns and the collaborative aspect. These limitations

highlight the need for a more refined batching approach that can accurately represent resource workflows.

In the next section, we will introduce a new batching approach that addresses some of these limitations and better fits the operational requirements. This new approach aims to improve the resource collaboration aspect.

6.3 Batching over Activity

In the previous section 6.2, we explained the batching over resource approach. In the current section we formalize another batching detection technique Bathing over Activity by presenting its mathematical definition and implementation steps. This method focuses on grouping events based on type of activity performed on a process stage (station) and their temporal proximity. Additionally, we provide provide an example of Batching over Activity on the Figure 6.9.

In addition to the visual representation of the Batching over Activity approach we provide BatchInstance node attributes description in the Table 6.3

Table 6.3: BatchInstance Attributes for Batching over Activity Approach Visualization

BatchInstance 1	
Batch_number	13974
Activity	Montaje
Earliest_timestamp	2022-03-29T13:18:00Z
Latest_timestamp	2022-03-29T13:29:00Z
Events_number	8
Kits	[HNB-TR.010-4.0, HUBU-TR.003-5.0, EXT-QUI.HARM_AZUL, EXT-QUI.012, HUBU-TR.014-1.0, DEP-QUI.TRA.STRYK-37.0, HNB-URO.001-5.0, HNB-TR.010-3.0]
Kits_number	8
Runs	[HNB-TR.010-4.0-43, HUBU-TR.003-5.0-20, HNB-URO.001-5.0-22, HNB-TR.010-3.0-40]
Users	[MCE, PN, VS, EH, SP]
Users_number	5

Based on the Figure 6.9 and Table 6.3 we can observe the parallel work of several resources on the same process stage, performing identical types of

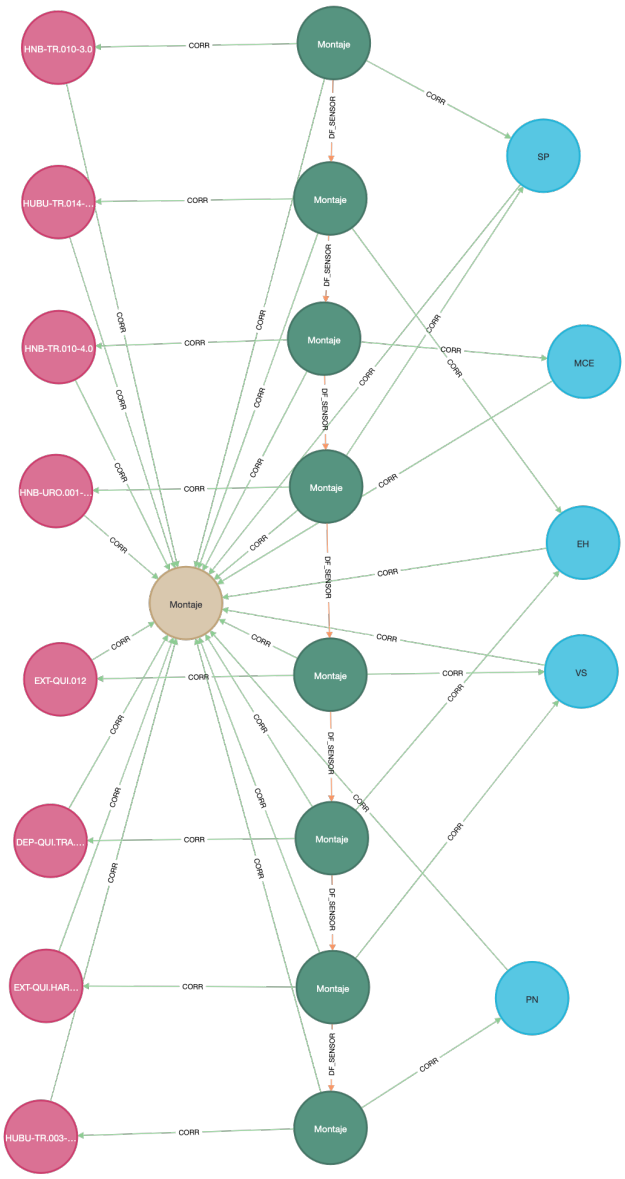

Figure 6.9: Example of Batching over Activity

tasks. In the visualization, green nodes labeled "Montaje" represent individual event nodes, each signifying an occurrence of the same activity. These nodes are connected with CORR relationships to pink nodes representing kits and blue nodes denoting resources, highlighting the involvement of various kits and resources in the Montaje operations. The DF_Sensor edges connecting these event nodes indicate that the kits underwent scanning during the "Montaje" process, while the CORR edges show the specific kits and

resources involved in each event.

This BatchInstance captures a group of events at the Montaje station, spanning an 11-minute window from the earliest event timestamp at 2022-03-29 13:18:00 to the latest at 2022-03-29 13:29:00. The batch comprises 8 events, involves 8 kits, includes 4 runs, and was facilitated by 5 users, demonstrating a batch operation.

6.3.1 Mathematical Definition of Batching over Activity

Definition 15. ***(Batch over Activity)*** *The mathematical formula for defining batches over activity based on the criteria mentioned in the Section 6.1 is as follows:*

$$Batch_{station}(e_i, e_j) = \begin{cases} 1 & \textit{if } activity(e_i) = activity(e_j) \\ & \textit{and } |timestamp(e_i) - timestamp(e_j)| < 5 \textit{ minutes} \\ 0 & \textit{otherwise} \end{cases}$$

Where: - e_i and e_j are consecutive events. - activity(e) represents the activity associated with the event. - timestamp(e) is the time at which the event occurred.

This formula helps in determining whether two events belong to the same batch based on their activities and the time difference between their occurrences.

6.3.2 Batching over Activity: Implementation Steps

The batching over activity process involves several steps:

1. **Assigning Batch Attributes:** Similarly to the batching over resource approach we need to associate each event node in the EKG to a specific batch. Thus, we introduce a "batch" attribute for each event node, assigned numeric value that identifies a specific batch of event group. The assignation performed in the following order:

   (a) **Sequential Evaluation:** Event nodes are sorted and assessed based on their `timestamp` attribute to maintain chronological order, alongside the `activity` attribute, ensuring that each event is associated with one type of work.

(b) **Grouping Criteria:** Each event node is compared with the preceding one to determine if they occur within a 5-minute window, as marked by the `timestamp`.

(c) **Batch Attribution:** Events that align under the grouping criteria are assigned the same batch number. If the criteria are not met, a new batch is started. This batch number is then recorded as the 'batch' attribute for each event node.

This step insures classifying the event nodes by batch.

2. **Creating BatchInstance Nodes and Connecting to Event Nodes:** Once the batch attributes are assigned to event nodes, we proceed with creation of BatchInstance nodes, that present these batches in the EKG. Each BatchInstance node represents the events that share the same resource and activity within the 5-minute window. A BatchInstance node for this type of batching includes attributes:

   (a) `Activity` – represents the activity (station on which batching occurs) of all associated events of a batch.

   (b) `Earliest_timestamp` – a timestamp of the first event in the batch, marking the start of the batch.

   (c) `Latest_timestamp` – a timestamp of the last event in the batch, marking the end of the batch.

   (d) `Events_number` – the number of events included in the batch, providing a quantitative measure of the batch size.

   (e) `Kits` – a list of kits included in the batch, detailing the specific items involved.

   (f) `Kits_number` – the number of kits included in the batch, indicating the diversity of items processed.

   (g) `Runs` – a list of runs associated with kits in the batch, reflecting the operational flow.

   (h) `Users` – a list of users who performed the events in the batch, capturing the human resources involved.

   (i) `Users_number` – the number of users who performed the events in the batch, highlighting the level of collaboration.

   Example of created BatchInstance node for batching over activity shown on the Figure 6.9 where the node colored in ligh-brown.

Following the logic from batching over activity resource, we proceed with connection BatchInstance nodes to Event nodes with correlation relationships (CORR) based on the shared batch number attribute. This edge in EKG ensures that all events included in a batch are associated with the corresponding BatchInstance. Events within the same batch meet the criteria established in the earlier section 6.1. On the Figure 6.9 we can observe green edge CORR that represent established correlated relationship between green Event nodes and a light-brown BatchInstance node.

3. **Establishing Relationships:** Finally, we create the other required correlation relationships (CORR) between BatchInstance nodes and Resource, and Kit nodes and add directly-follow relationships between BatchInstance nodes:

   (a) **Correlated Relationships:** Each BatchInstance node is linked to the corresponding Resource and Kit nodes to accurately represent their interactions within a batch. Specifically, BatchInstance nodes are connected to Resource nodes based on the shared resource's system ID. It is important to mention that in the batching over activity approach provide a possibly to connect one BatchInstance with several Resource nodes as if we capture simultaneous work of resources on one process stage in a particular time frame. Additionally we establish correlation relationships between BatchInstance nodes and Kit nodes by associating through shared kit identifier. Similarly, we can have multiple correlated relationships from one BatchInstance node to multiple kits. This ensures that all resources and kits involved in the batch are properly represented. In the provided Figure 6.9, we can observe this type of relationship as a green edge named CORR between pink nodes representing Kit nodes, while the blue node representing the Resource node and light-brown BatchInstance nodes.

   (b) **Directly-Follow Relationships:** This process includes creating two types of directly-follow relationships DF_BATCH_KIT and DF_BATCH_RESOURCE. These relationships are constructed following the methodology of directly-follows aggregation outlined by Klijn, Mannhardt, and Fahland [16] mentioned in the Section 3.4.1 with additional extension. By establishing these directly-follow relationships, we create a detailed view of the workflow within the sterilization process of kits and resource on the batching aggregation level. Details on changes for each of the edge presented

below:

i. **DF_BATCH_KIT:** The relationship has been enhanced by adding attributes `kitId` and `runId` to each edge. This adjustment is crucial for identifying which edge corresponds to each kit, especially in scenarios where multiple incoming and outgoing edges exist. By specifying these attributes, we clarify the connection for each kit, ensuring that its path is followed.

ii. **DF_BATCH_RESOURCE:** The enhancements of this type of relationship focus on refining how we capture the complex nature of resource tasks. These enhancements include the addition of specific attributes to the relationship edges:

   A. `SysId:` Including the resource's identifier to specify which resource performed the activities.

   B. `Count:` Counting how many times this edge (relationship) occurs, which helps in understanding the frequency and iterative manner of the work performed by the resource.

   C. `Order:` Determining the order of these relationships to capture the sequence of operations on a particular day. This helps in visualizing the daily workflow of the resource.

   D. `Outgoing_order:` Capturing the sequence of outgoing edges from a source BatchInstance node. This is useful for tracking the multiple transitions a resource might make from a single batch.

By specifying these attributes, we clarify the connection for each resource and capture complex workflow, ensuring that a path is followed.

We provide additional Figure to illustrate implemented edges 6.10. These relationships for Resources are represented as blue edges DF_BATCH_RESOURCE named with associated resource system identifier connecting two light-brown BatchInstance nodes, illustrating how batches are sequentially linkedfollowing resource path. Additionally, orange edges DF_BATCH_KIT for capturing flow of kits between BacthInstances.

With this final step, we established the required relationships within the Event Knowledge Graph, allowing us to view a fully aggregated subgraph that illustrates batching over activity.

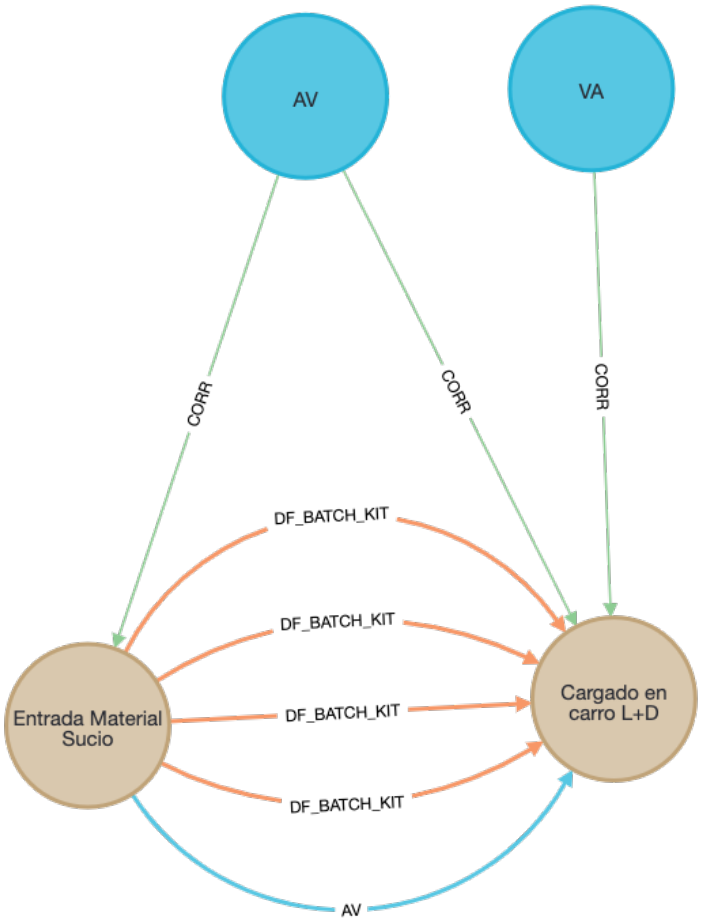

Figure 6.10: Illustration of DF_BATCH_RESOURCE edge for "AV" resource

In summary, the implementation steps of Batching over Activity detection methodology in this subsection provide an approach for grouping event nodes into BatchInstance nodes and enriching EKG. Initially, it groups event nodes into BatchInstance nodes. It then establishes relationships between the batches and their corresponding events, resources and kits and after within these batches. By linking these components, we create a subgraph within the EKG, enabling visualization of the batching processes, as demonstrated on Figure 6.9 and Figure 6.10.

6.3.3 Batching over Activity: Implementation Results and Exploration

In this section, we aim to evaluate the outcomes of the Batch over Activity approach. We begin by reviewing the key implementation statistics and then proceed to a detailed exploration of the resulting framework.

The implementation process led to the following results:

Table 6.4: Implementation Overview for Batch over Activity Approach

Metric	**Count**
BatchInstance Nodes	16,947
Correlation Relationships	425,096
Directly-Follow Relationships for Kits	139,578
Directly-Follow Relationships for Resources	61,786

It is evident that the space of nodes and edges is significantly smaller compared to the batching over resource approach mentioned in the Section 6.2.3.

Improved Batch Size The batch over activity approach provides a better representation of batches, capturing significantly more events within each batch. This approach aligns more closely with the intended concept of batching, as compared to the batch over resource method. The figure 6.11 illustrates this distribution.

Based on the analysis, 20.64% of the total batches consist of a single event, while the remaining 79.26% include two or more events. This is a notable improvement compared to the batch over resource approach, where a larger proportion (58%) of batches contained only one event. The batch over activity method thus offers a more cohesive and meaningful aggregation of events.

Solved Issue of Concurrent Resource Utilization The previous batching over resource approach had a significant limitation in representing situations where multiple resources worked on the same activity at the same time mentioned in the Section 6.2.3. This led to fragmented batches that did not accurately reflect the collaboration between resources.

The batching over activity method resolves this issue by linking resources performing the same task within a specific time frame to a single BatchInstance. This ensures that all concurrent activities are correctly grouped,

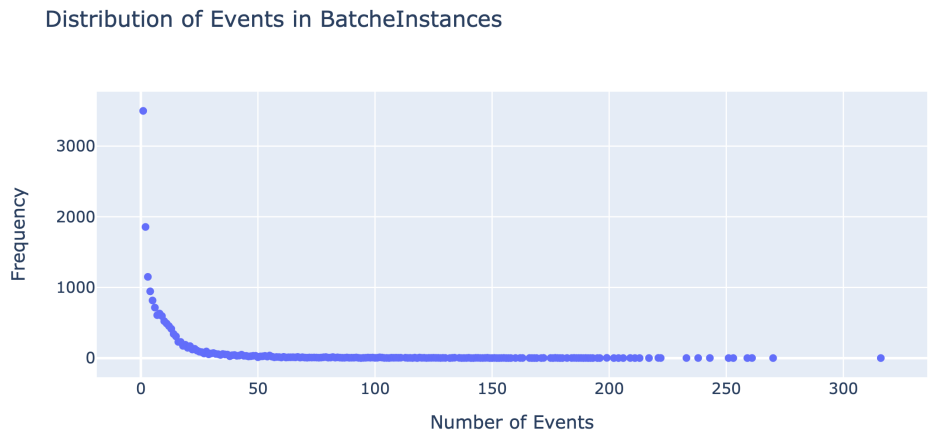

Figure 6.11: Distribution of Events in BatcheInstances for Batching over Activity Approach

providing a more accurate and complete representation of the work being done.

We provide a bar chart on the Figure 6.12 representing the distribution of BatchInstances based on the number of resources involved. The x-axis indicates the number of resources associated with each BatchInstance, while the y-axis shows the frequency, or the number of BatchInstances that fall into each category.

From the chart, it is clear that the majority of BatchInstances (13,033 instances) involve only a single resource. As the number of resources increases, the frequency of BatchInstances decreases significantly, with 1,531 instances involving two resources, and progressively fewer instances as the number of resources increases further.

The Figure 6.12 reveals that while the majority of BatchInstances are completed by a single resource, a significant 23% involve collaboration among multiple resources. This highlights the importance of considering both individual and collective work patterns in the analysis.

From activity perspective we analysed where more often is performed parallel work. The bar chart presented on the Figure 6.13 covers nine all distinct activities where multiple resources are involved in the same BatchInstance.

The majority of BatchInstances with multiple resources occur in activities like "Montaje," "Entrada Material Sucio," and "Producción montada." These activities show significant collaboration, particularly when two or more resources are involved. As the number of resources increases, the frequency of

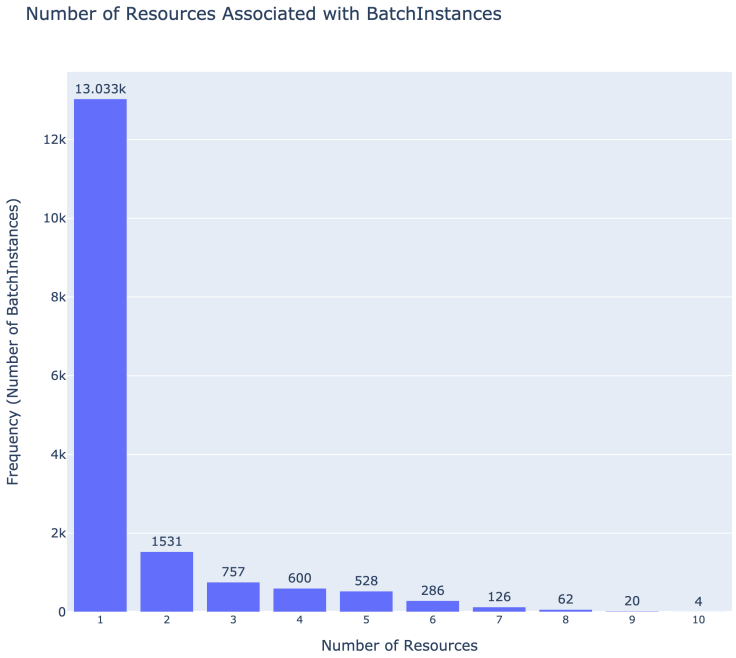

Figure 6.12: Distribution of BatchInstances Based on the Number of Resources

BatchInstances decreases, but simultaneous work still occurs across various activities, indicating the collaborative nature of these tasks. This distribution highlights the importance of understanding how resources are allocated across different activities based on batching over activity approach.

Understanding Resources Allocation within Batching over Activity Approach The goal of this particular analysis is to gain an understanding of how resources are allocated across various activities within the operational workflow. To achieve this, we utilize the Figure 6.14. This graph is designed to illustrate how batches over activities essentially group multiple resource-specific batches into cohesive units. In other words, each batch at the activity level represents a collection of smaller batches from different resources working on the same activity. In the visualization, the y-axis shows the count of BatchInstances, while the x-axis lists the individual resources involved. The different colors indicate various activities, showing how each resource contributes to specific tasks. For instance, the resource "MCE" is involved in activities such as "Producción montada" and "Montaje," indicating its role in these key processes. Similarly, resources like "PN" and "SM" show

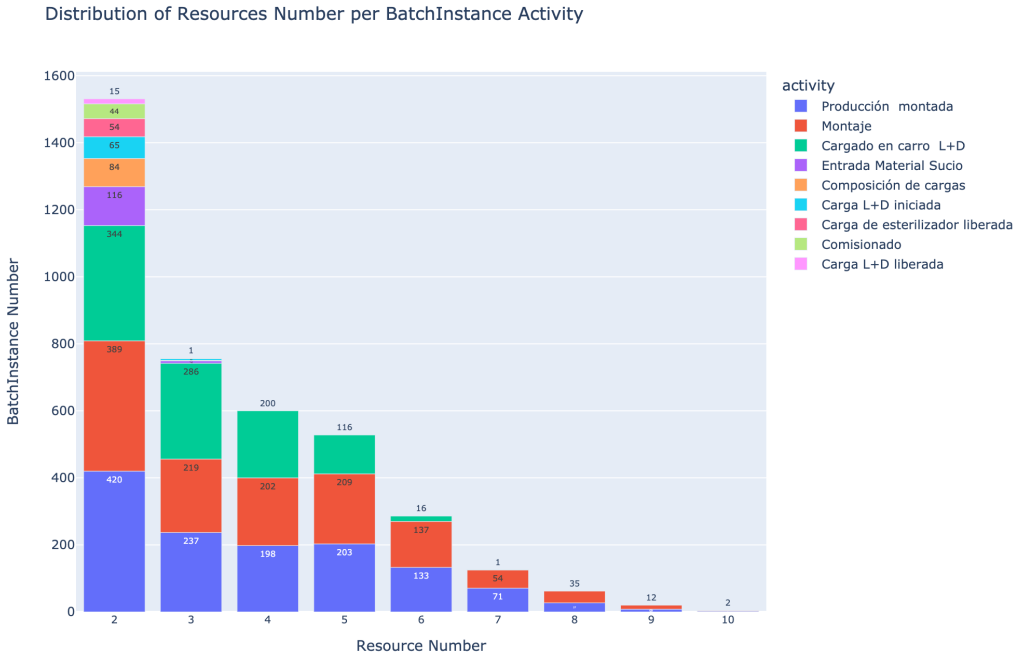

Figure 6.13: Distribution of Resources Number per BatchInstance Activity

contributions across multiple activities, reflecting their involvement in the overall workflow.

Other resources, like "ER," "BM," and "EH," also participate in a range of activities, although their involvement is slightly less than that of "MCE," "PN," and "SM." This pattern suggests that these resources may have more specialized roles or that their efforts are spread across different tasks. Meanwhile, resources such as "LI," "VA," and "MR" are involved in fewer BatchInstances but still contribute to various tasks, indicating their roles in specific, possibly more specialized, activities.

By capturing the collective contributions of multiple resources to a single activity, it provides a view of resource allocation and workload distribution.

Iterative Pattern Similar to the Batching over Resource approach, the Batching over Activity approach also exhibits the pattern described in Section 6.2.3. This pattern occurs in the workflow of resources that begin their work within a specific BatchInstance, briefly switch to another type of work in a different BatchInstance, and then return to the initial BatchInstance.

We present the Figure 6.15 that demonstrates the pattern. We can ob-

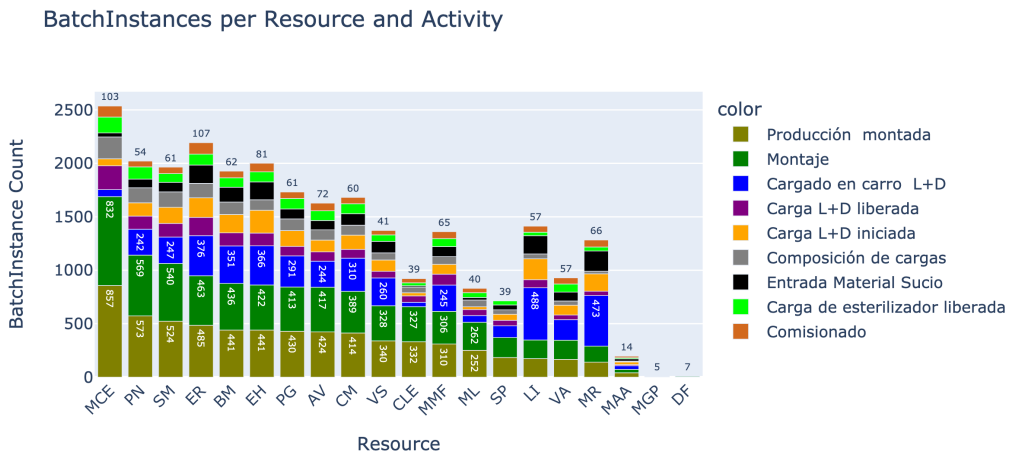

Figure 6.14: BatchInstances per Resource and Activity

serve an iterative pattern of batches performed by the resource "MMF" within the workflow. The green nodes represent different events associated with two main activities: "Producción montada" and "Montaje." The blue node represents the resource "MMF," and the light brown nodes represent BatchInstances where the resource revisits previous activities.

The DF_RESOURCE edges between the event nodes indicate the sequence of activities. The DF_BATCH_RESOURCE edges reflect the transitions between BatchInstances correlated to this event nodes. The arrows show that "MMF" alternates between "Producción montada" and "Montaje," consistently returning to "Producción montada" after a brief engagement with "Montaje."

This pattern demonstrates how the resource moves back and forth between these two batches, highlighting the iterative nature of the workflow.

To analyse the pattern, we constructed a query was constructed. This query targets sequences of three BatchInstance nodes that are connected by DF_BATCH_RESOURCE edges, ensuring that the BatchInstance nodes of these sequences are in correlated relationships with one resource. This linkage is validated by matching the sysId of the resource with the users attribute in the BatchInstance nodes and ensuring that the sysId aligns with the DF_BATCH_RESOURCE edges.

The query is designed to locate sequences where the first and third nodes is the same BatchInstance node. The middle node in the sequence must have a different activity attribute, representing a temporary shift in the resource's focus. The query includes a condition that compares the order attribute of

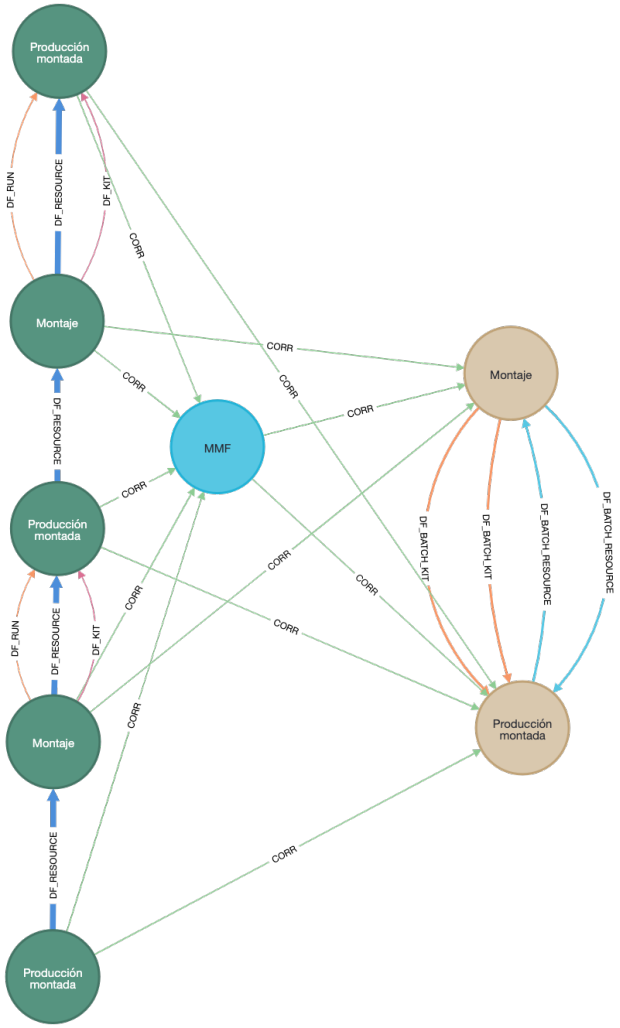

Figure 6.15: Iterative Pattern of Batches Performed by Resource "MMF"

the edges between the BatchInstance nodes. Specifically, it ensures that the order attribute of the edge connecting the first BatchInstance node to the second (r.order) is less than the order attribute of the edge connecting the second BatchInstance node back to the first (r1.order). This condition confirms that the sequence of BatchInstances is processed in the correct chronological order, ensuring that the transition from the first activity to the second happens before the resource returns to the initial activity.

The result is a distinct list of activity sequences, along with the frequency of their occurrence, ordered by how often these patterns appear and presented on the Figure 6.16.

In this chart, the x-axis represents different sequences of activities, while

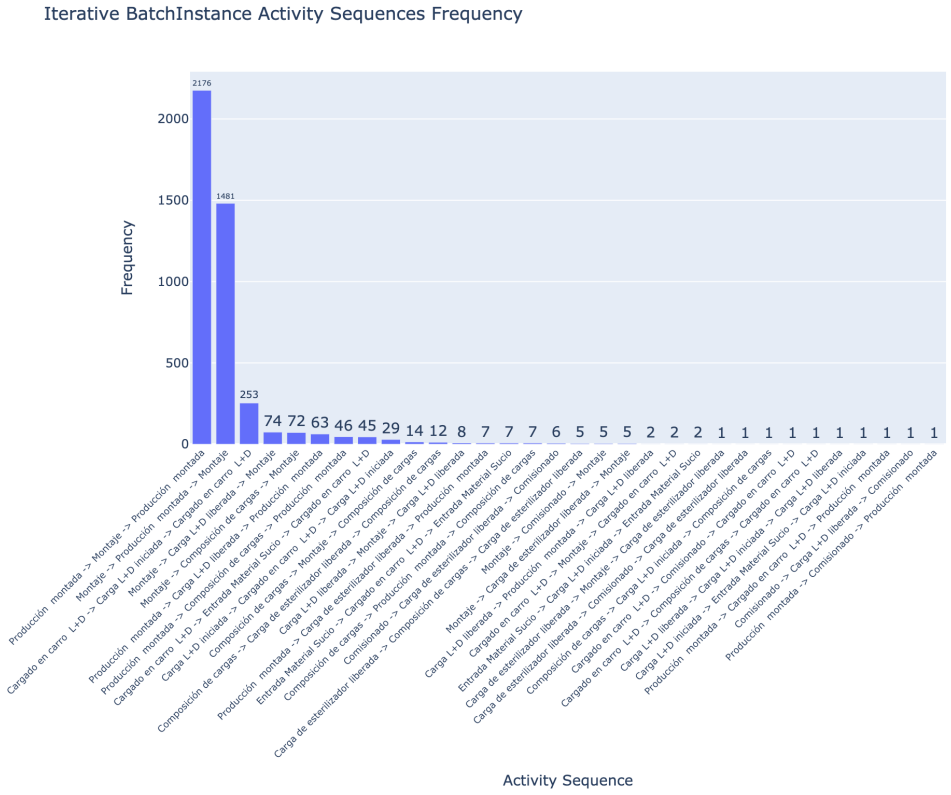

Figure 6.16: Patterns of Resources Returning to Previous BatchInstances

the y-axis indicates the frequency with which each sequence occurs. Each sequence is a pattern where a resource performs an initial batch, switches to a different batch, and then returns to the original batch back.

The most frequent sequence, "Producción montada -> Montaje -> Producción montada" occurs 2,176 times, highlighting that this pattern of returning to the initial batch after a brief switch to another is common. The second most frequent pattern, "Montaje - > Producción montada - > Montaje," appears 1,481 times, further emphasizing the iterative nature of these activities within the workflow.

As we move to less frequent sequences, such as "Carga L+D iniciada - > Cargado en carro L+D - > Carga L+D iniciada" and others, the frequency drops significantly. Despite being infrequent, these rare sequences—such as those appearing only once or a few times—demonstrate that even uncommon iterative behaviors exist. This suggests that there are instances of variability in how tasks are revisited.

This visualization underscores the recurrence of certain workflows and

helps identify where resources are frequently returning to previous tasks, which can be important for understanding resource behaviour.

Assessing Proposed Batching over Activity Batching over activity in comparison to batching over resource offers several advantages:

1. **Reduced BatchInstance and Edges Space:** Batching over activity reduces the number of BatchInstance nodes and edges, creating a more streamlined data structure. This makes data storage and retrieval more efficient for further analysis.

2. **Improves Batch Size:** This approach results in larger, more cohesive batches, capturing more events per batch compared to the batching over resource method.

3. **Solved the Issue of Concurrent Resource Utilization:** Batching over activity accurately represents scenarios where multiple resources work simultaneously on the same activity. This resolves the limitations of the batching over resource method, ensuring that collaborative work is properly reflected.

4. **Enhanced Resource Tracking from Activity Perspective:** By focusing on specific activities rather than resource involvement across tasks, this method allows for more precise tracking of resource allocation and performance.

5. **Improved Data Clarity:** The method's structured grouping of events reduces data clutter, making it easier to identify patterns and trends, and supports better decision-making.

However, while batching over activity provides better batching detection logic, it is still not sufficient for the analysis of resource workflow and working patterns. The current batching approach still falls in fully capturing resource workflows, particularly in addressing the issue of resources returning to previous BatchInstances. This indicates that the current level of data aggregation is insufficient. To overcome this limitation, it is necessary to perform a higher level of aggregation.

In the following chapter, we will define and provide the implementation of the high-level aggregation that aims to solve the limitation of Batching over activity and offer a better framework for resource analysis.

Chapter 7

High-Level Batching

Building on the limitations of the batching over activity approach discussed in Section 6.3, it becomes evident that while this method effectively models collaboration within a batch, it also introduces challenges related to iterative behavior. When using batching over activity, resources switch between tasks, which can cause batches to overlap temporarily. This overlap makes it challenging to track and model what an individual is doing over a period of time accurately. To resolve this issue, we propose high-level aggregation of batches for each resource and a day.

The primary goal of high-level aggregation is to address the iterative behavior limitations of the batching over activity approach and to offer a more effective framework for resource analysis by creating a high-level event log.

To this end, this chapter first explores the motivation for high-level aggregation in Section 7.1. We then proceed with defining high-level batching and outlining the implementation steps and results of it in the Section 7.2.

7.1 Motivation and Requirements for High-Level Batch Aggregation

The motivation for implementing high-level batching arises from the limitations observed in the batching over activity approach in the Section 6.3. While the activity-based batching method offers a clear and structured grouping of events based on activities and temporal proximity, it falls short in capturing the iterative and repetitive nature of resource workflows. That cause a temporal overlap between resource's batches. Additionally, it also fails to capture the unique resource input into BatchInstances with several resources.

To illustrate these challenges more concretely, consider the following example on the Figure 7.1.

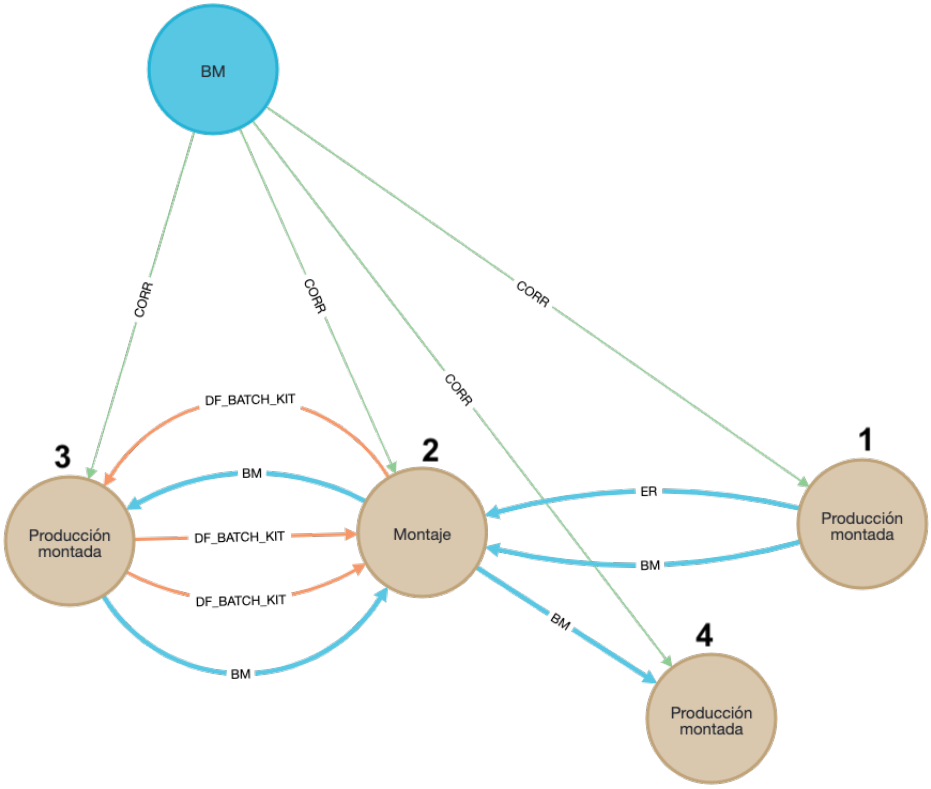

Figure 7.1: Example of Overlapping Batches and Iterative Patterns

The graph in Figure 7.1 provides an illustrative example of overlapping batches for resource "BM", showcasing a segment of BM's workflow on 02-08-2022 between 20:59 and 21:32. In this graph, the light brown nodes represent BatchInstances, we numbered them to provide a clear view on the resource's workflow. Numbering of nodes based determined by the `order` attribute of the blue DF_BATCH_RESOURCE edges that are specific to resource "BM" (edges named BM).

This example highlights an iterative pattern between BatchInstance nodes 2 and 3, where resource BM repeatedly switches between BatchInstances of a particular type of work. The detailed information for these nodes is presented in Table 7.1.

Resource "BM" performs in an iterative pattern twice between these two nodes (2 and 3), as indicated by the `count` attributes of the associated DF_BATCH_RESOURCE edges. Further, BM exits this iterative pattern by transitioning from node 2 to node 4, demonstrating a shift in workflow.

Table 7.1: BatchInstances Attributes for Resource BM's Iterative Pattern

BatchInstance 2	
Activity	Montaje
Batch_number	12466
Earliest_timestamp	"2022-02-08T21:04:00Z"
Latest_timestamp	"2022-02-08T21:12:00Z"
Events_number	4
Kits	[EQP-QUI.CP.OPT-1.0, EQP-QUI.OPT-17.0, HNB-CG.003-5.0, HNB-CPL.007-1.0]
Kits_number	4
Runs	[EQP-QUI.CP.OPT-1.0-1, EQP-QUI.OPT-17.0-1, HNB-CG.003-5.0-7, HNB-CPL.007-1.0-0]
Users	[BM, ER]
Users_number	2
BatchInstance 3	
Activity	Producción montada
Batch_number	15300
Earliest_timestamp	"2022-02-08T21:06:00Z"
Latest_timestamp	"2022-02-08T21:12:00Z"
Events_number	4
Kits	[EQP-QUI.OPT-17.0, EQP-QUI.CP.OPT-1.0, HNB-CG.003-5.0, HUBU-ORL.021-3.0]
Kits_number	4
Runs	[EQP-QUI.OPT-17.0-1, EQP-QUI.CP.OPT-1.0-1, HNB-CG.003-5.0-7, HUBU-ORL.021-3.0-2]
Users	[BM, CLE]
Users_number	2

Notably, the BatchInstance nodes involved in this iterative pattern also include contributions from other resources, such as ER and CLE.

To address these challenges, we propose implementing high-level aggregation. By combining BatchInstances into high-level batches. High-level aggregation also aims to create a sequence of high-level batches for each resource, organized separately within a day, resulting in a unified high-level log that captures the overall workflow of resources within Company C's sterilization process.

7.1.1 Batching Patterns that Lead to High-Level Aggregation

The first step in understanding how to correctly perform high-level aggregation is to identify the patterns that lead to this aggregation. These patterns provide a structured approach to how events and activities are grouped within BatchInstances, revealing the iterative behaviour of resource workflows. We define three primary patterns that form the basis for high-level batching:

1. **Regular Pattern:** A regular pattern describes the behavior of a resource that performs events belonging to one BatchInstance and then moves on to perform events of a subsequent BatchInstance. This pattern reflects a straightforward and linear progression of tasks, providing a clear view of sequential activities handled by a resource.

   In Figure 7.2, we can observe the regular pattern for resource AV. The resource works sequentially between the activities "Composición de cargas" and "Producción montada," moving from one BatchInstance to the next without returning to the initial BatchInstance.

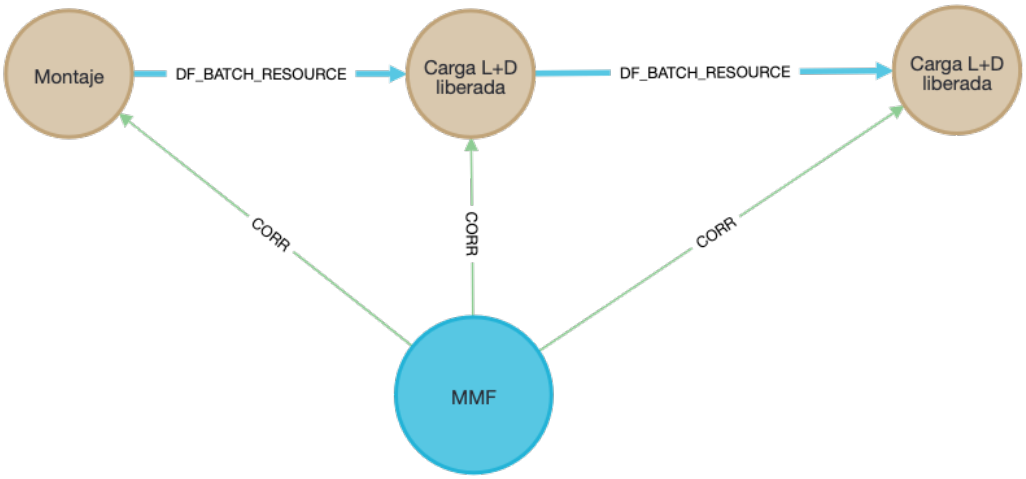

Figure 7.2: Regular Pattern

2. **Iterative Pattern:** An iterative pattern describes the behavior of a resource that performs events belonging to one BatchInstance, then moves on to perform events of a subsequent BatchInstance, and finally returns to perform events of the initial BatchInstance. This pattern captures the repetitive nature of some workflows, where a resource frequently revisits previous tasks or BatchInstances.

Figure 7.3 demonstrates the iterative pattern for resource MMF. The resource iterates between the activities "Montaje" and "Carga L+D liberada," performing events in one BatchInstance, then in another, and eventually returning to the initial BatchInstance.

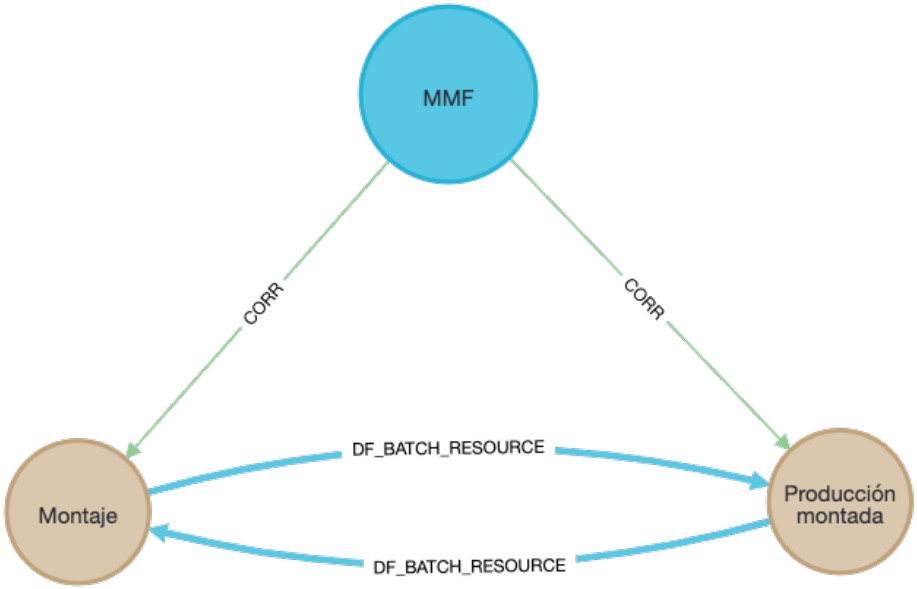

Figure 7.3: Iterative Pattern

3. **Complex Iterative Pattern:** A complex iterative pattern describes the behavior of a resource that performs an iterative pattern, then temporarily shifts to handle a different BatchInstance, and subsequently resumes the original iterative pattern. This pattern highlights the complex and multi-faceted nature of resource activities, showing how resources may juggle multiple tasks and revisit them as necessary. Complex iterative pattern considered within a depth of three nodes. Extending this depth further could risk incorporating over 30% of the entire process, given that the sterilization workflow consists of nine distinct stages.

   In Figure 7.4, we observe resource AV managing multiple BatchInstances by following the DF_BATCH_RESOURCE edges labeled "AV.". The resource shows complex iterative behavior between the activities "Montaje" and "Producción montada," temporarily shifting to handle another BatchInstance "Composicion de cargas" and then resuming the original iterative pattern. This illustrates the resource's capability to manage complex workflows involving multiple tasks and revisits.

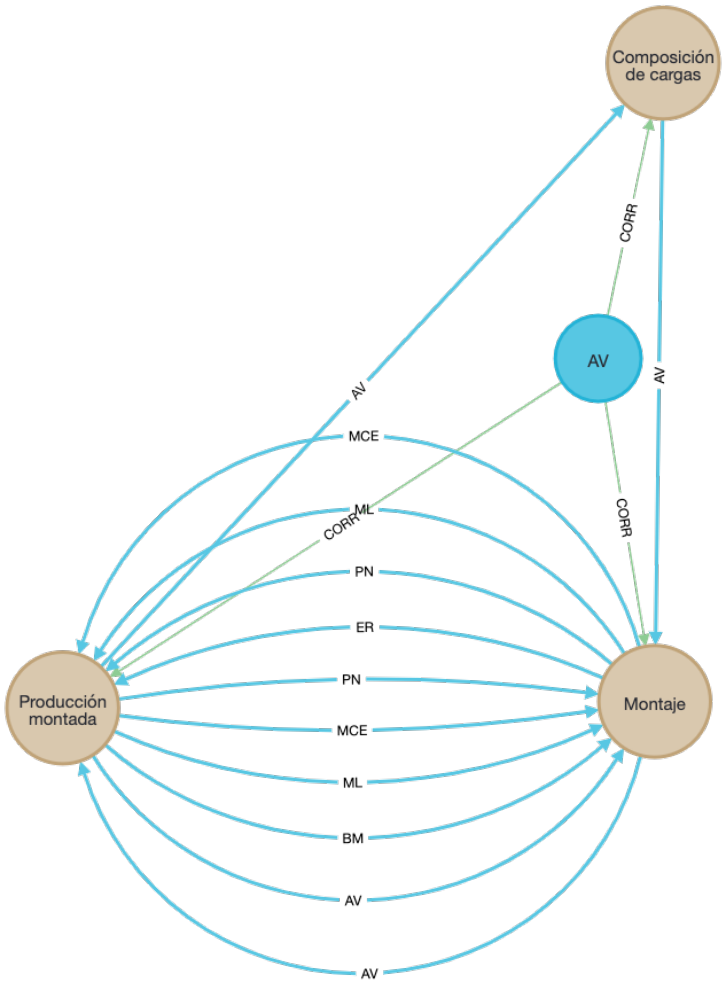

Figure 7.4: Complex Iterative Pattern

Identifying these patterns is important for further high-level batch aggregation implementation, as they provide method for grouping BatchInstances according to resource workflows.

7.1.2 Requirements for High-level Batching Aggregation

To implement high-level batching, two requirements must be met:

1. **Eliminating Iterative Behaviors and Temporal Overlaps:** High-level aggregation must be capable of capturing the patterns defined in the previous Subsection 7.1.1: regular, iterative, and complex iterative

patterns considering the temporal overlaps caused by resources switching between batches (BatchInstances). This capability ensures that the aggregation reflects the actual workflow dynamics and provides a basis high-level log creation and further resource analysis

2. **Isolating Unique Contributions:** This involvement of multiple resources within the same BatchInstance nodes needs to be considered in the construction of a high-level batch structure. To accurately represent each resource's workflow per day, it is important to isolate the unique contributions of a resource within these BatchInstances before aggregating them into a high-level batch.

These requirements are considered for implementation. We present implementation steps in the next section 7.2

7.2 High-Level Batching: Definitions and Implementation Steps

In this section, we define high-level batches mathematically and outline the detailed implementation methodology. We also present the implementation results of high-level batching and how they are represented within the Event Knowledge Graph (EKG).

7.2.1 Mathematical Definition of High-Level Batches

The mathematical definitions for high-level batches are as follows:

Definition 16. ***(Singular High-Level Batch)*** *A Singular High-Level Batch corresponds to a regular pattern and is defined as:*

$$HLB_{singular}(r, d) = \{b_i\}$$

where r represents a particular resource, d is a specific date, and b_i is a single BatchInstance correlated to the resource on that date.

Definition 17. ***(Joint High-Level Batch)*** *A Joint High-Level Batch corresponds to an iterative pattern involving two BatchInstances and is defined as:*

$$HLB_{joint}(r, d) = \{b_i, b_j\}$$

where r represents a particular resource, d is a specific date, and b_i, b_j are two BatchInstances correlated to the resource on that date.

This indicates a workflow where the resource revisits a previous task.

Definition 18. ***(Complex Joint High-Level Batch)*** *A Complex Joint High-Level Batch corresponds to a complex iterative pattern involving three BatchInstances and is defined as:*

$$HLB_{complex\_joint}(r, d) = \{b_i, b_j, b_k\}$$

where r represents a particular resource, d is a specific date, and b_i, b_j, b_k are three BatchInstances correlated to the resource on that date.

This reflects a workflow where the resource revisits tasks, with potential temporary shifts to another task before returning to the original ones.

7.2.2 Implementation of High-Level Batching Aggregation

The high-level batching process involves several steps:

The primary challenge addressed in this step is associating each BatchInstance node for a resource per day in the Event Knowledge Graph (EKG) with a specific high-level batch. This process involves creating high-level batches and the necessary edges to represent relationships between these batches.

1. **Creating High-Level Batch Nodes:**

   (a) **Fetching Resource Paths per Day:** Begin by retrieving all paths of BatchInstances for a resource in one day. These paths are analyzed based on the `order` attribute of the `DF_BATCH_RESOURCE` edges, which indicates the sequence of BatchInstances.

   (b) **Singular High-Level Batch Creation:** For linear paths where a resource transitions from one BatchInstance to another without iterations and demonstrate a regular pattern, created a singular high-level batch node for each of BatchInstance.

   (c) **Joint High-Level Batch Node Creation:** If an iterative pattern is identified, where a resource returns to a previous BatchInstance after an intermediate task, merged two BatchInstance nodes into a joint high-level batch.

   (d) **Complex Joint High-Level Batch Creation:** For complex patterns where a resource temporarily shifts to handle additional BatchInstances before returning to the original iterative pattern, created a complex joint high-level batch. This high-level batch aggregates three BatchInstances.

   Each HighLevelBatch node includes the following attributes:

i. `Activity_name` – The list of activities from the BatchInstances that are included in the high-level batch.

ii. `Corr_batch_numbers` – A list of BatchInstance numbers that are grouped together to form the high-level batch.

iii. `Date` – The date on which the high-level batch was created.

iv. `Start_timestamp` – The earliest timestamp of events within the BatchInstances that are included in the high-level batch.

v. `End_timestamp` – The latest timestamp of events within the BatchInstances that are part of the high-level batch.

vi. `Number_of_events` – The total number of events within the BatchInstances that are included in the high-level batch.

vii. `SysId` – The system ID of the resource associated with the high-level batch.

viii. `WorkTogether` – Indicates whether multiple resources collaborated within the high-level batch. This attribute is set to true if at least one BatchInstance used to form the high-level batch involved two or more resources, as indicated by the `Users_number` attribute of the BatchInstances.

Created high-level bath nodes represent a single or multiple joined BatchInstances for a resource on a particular day.

2. **Establishing Relationships:**

   (a) **Correlated Relation to BatchInstances:** We create edges (CORR) between BatchInstance nodes and created high-level batche nodes based on the shared attribute of `Users_number` in BatchInstance nodes and `Corr_batch_numbers` in high-level batch nodes.

      i. **Connecting High-Level Batches to BatchInstances:** We connect high-level batch nodes to their respective BatchInstances based on the batch numbers. This linkage ensures that all BatchInstances involved in a high-level batch are associated correctly.

      ii. **Connecting High-Level Batches to Resources:** We establish edges between high-level batch nodes to their corresponding resource nodes based on the shared attribute of resource system identifier.

      iii. **Connecting High-Level Batches to Events:** We create correlated relationships between events and high-level batches on the shared batch number attribute and correlation to the same resource node.

(b) **Directly-Following Relations:** We create directly-follow edges (DF_HIGH_LEVEL_BATCH) between high-level batches of a particular resource on a specific date using the `order` attribute of DF_BATCH_RESOURCE between BatchInstances associated with high-level batches to capture the sequence of operations.

This detailed mapping of resource movements through various BatchInstances helps identify high-level workflow of a resource.

In summary, the implementation ensures that high-level batches are formed based on identified patterns mentioned in the Section 7.1.1, providing a structured aggregation of BatchInstances nodes and enriching EKG.

7.2.3 High-Level Batching: Implementation Results and Representation in the EKG

Current section is dedicated to explore the implementation results and representation of high-level aggregation in the EKG. We begin by reviewing the key implementation statistics. The Table 7.2 presents the key implementation statistics, including the number of high-level batch nodes created, the correlated relationships established, and the directly follow relationships formed.

Table 7.2: Implementation Statistics for High-Level Batch Nodes

Metric	**Count**
High-Level Batch Nodes Created	22,023
Correlated Relationships	267,769
Directly Follow Relationships	42,830

In addition, we present visualisation of each type of high-level batch and associated patter on the batch level. These visualizations help illustrate the different types of high-level batches formed during the aggregation process, based on the regular, iterative, and complex iterative patterns:

1. **Singular High-Level Batch:** This type of batch corresponds to a regular pattern and is correlated to only one BatchInstance of a particular resource on a specific date. It represents a straightforward aggregation of events handled by a resource in a linear sequence. The Figure 7.5 represent the singular High-Level Batches and correlated BatchInstances.

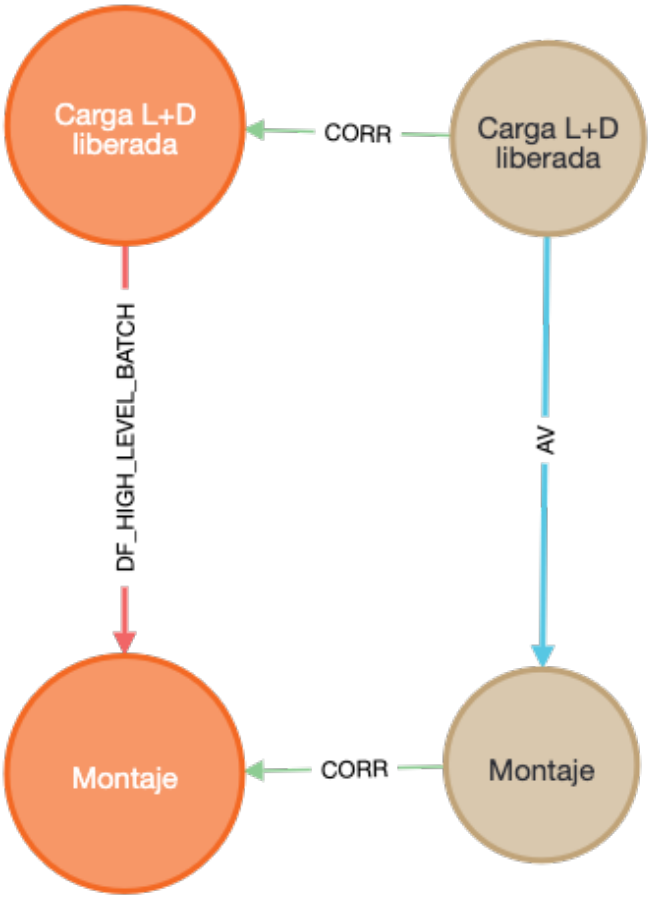

Figure 7.5: Singular High-Level Batch

2. **Joint High-Level Batch:** This type of batch corresponds to an iterative pattern and is correlated to two BatchInstances of a particular resource on a specific date. It captures the iterative nature of the resource's workflow, showing multiple interactions with two different batches. The Figure 7.6 represent the Joint High-Level Batche and correlated BatchInstances.

3. **Complex Joint High-Level Batch:** This type of batch corresponds to a complex iterative pattern and is correlated to three BatchInstances of a particular resource on a specific date. It represents the aggregation, reflecting the multi-tasking and revisiting nature of the resource's activities. The Figure 7.7 represent the Complex Joint High-Level Batch and correlated BatchInstances.

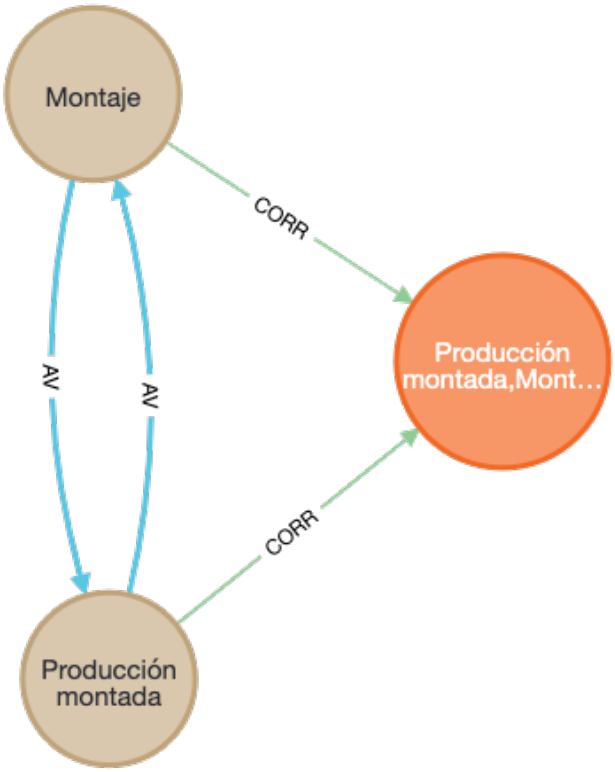

Figure 7.6: Joint High-Level Batch

We provide an additional visualization on the Figure 7.8 that offers a generalized view of the high-level aggregation process.

The figure illustrates the workflow of the "AV" resource on 03-03-2022, from 08:05 until 11:09. The orange nodes represent high-level batches, which are connected by DF_High_Level_Batch edges, demonstrating the resource's workflow at a high level. The light-brown nodes represent the associated BatchInstances, connected by DF_BATCH_RESOURCE edges labeled "AV," showing the resource's workflow at the batch level. The visualization highlights that the workflow primarily consists of singular high-level batches, with only two joint high-level batches. This joint batches captures the iterative pattern between the "Producción Montada" and "Montaje" BatchInstance nodes from the batching over activity approach.

In this chapter, we addressed the limitations of the batching over activity

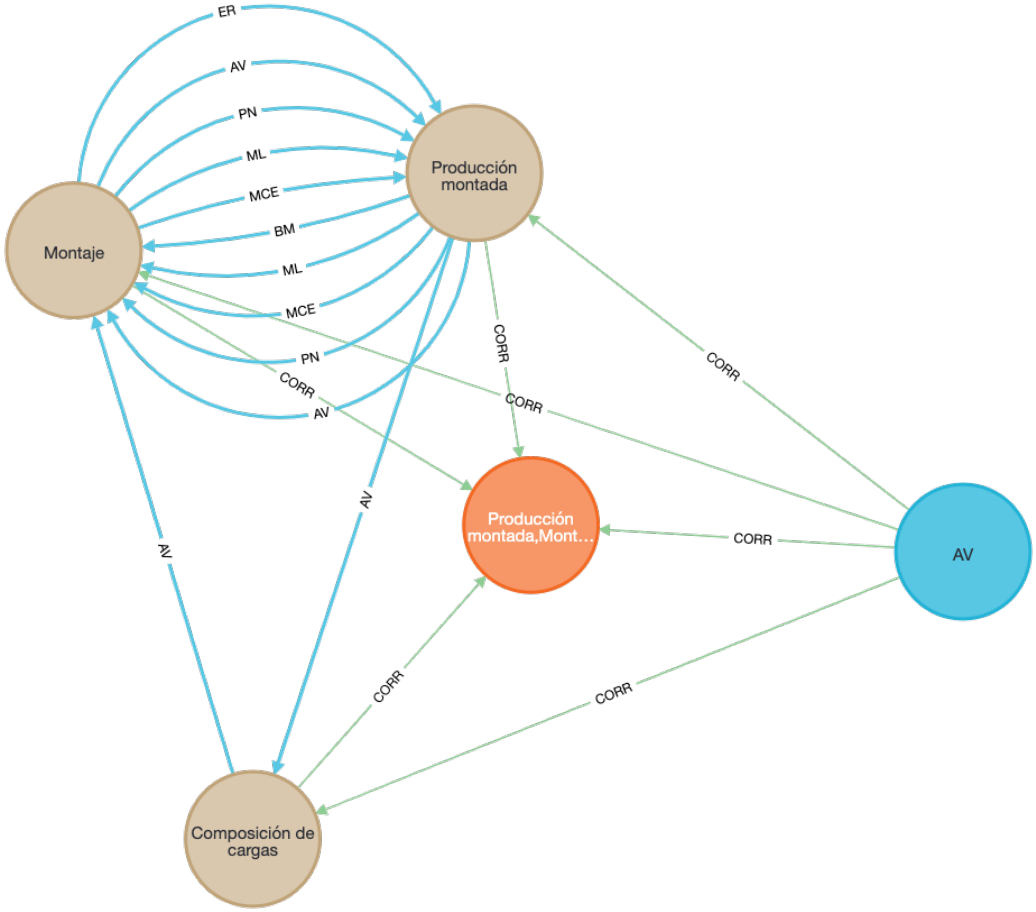

Figure 7.7: Complex Joint High-Level Batch

approach, particularly its shortcomings in capturing iterative behaviors with temporarily overlapping batches and modeling resource workflows over time. By introducing high-level batching, we were able to aggregate BatchInstances into high-level structures that represent the complexity of resource activities. This approach allowed us to identify and categorize key patterns—regular, iterative, and complex iterative—that are important for understanding the generalized flow of resources within the sterilization process at Company C. The implementation looked at complex iterative behaviors within a depth of three nodes. Going beyond this depth could include more than 30% of the entire process, given that the sterilization workflow has nine stages.

With the creation of over 22,000 high-level batch nodes and the establishment of nearly 310,000 relationships, we provided a foundation for the next phase of the thesis research. With the high-level batch aggregation

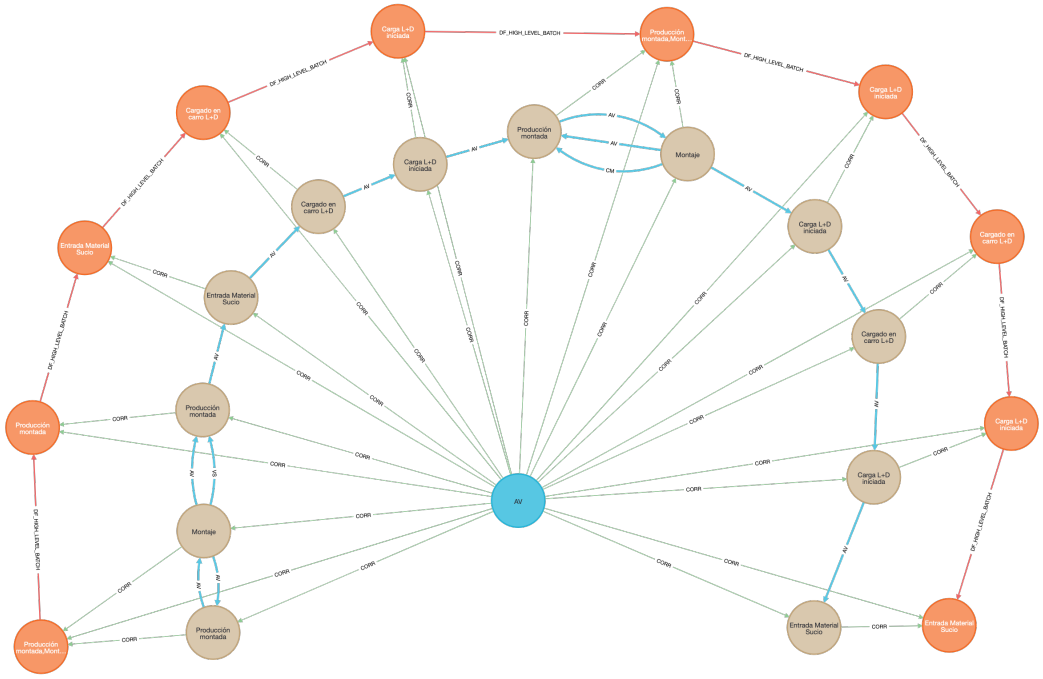

Figure 7.8: "AV" Resource High-Level Workflow

completed in this chapter, we are now prepared to form the high-level event log, which will serve as the basis for our subsequent resource analysis.

Chapter 8

High-Level Resource Behavioral Analysis

We have established high-level batches in the Chapter 7 that represent the resource flow, and based on these, we aim to achieve main goal of the thesis by analyzing resource working behavior and patterns. High-level batches provide a comprehensive view of how resources function across the entire workflow, capturing both their aggregated activities and transitions between tasks.

The primary method is a detailed analysis of resource behavior and patterns using a series of high-level events constructed from defined high-level batches. These high-level events reflect the daily activities of each resource, enabling a structured examination of their behavior throughout the workday. Thus, this log will allow to examine how resources function over time and identify patterns within their workflows.

We begin by forming the high-level event log through high-level batch aggregation, detailed in Section 8.1. Next, in Section 8.2, we analyze resource working behavior using the high-level event log from a workload perspective. Finally, in Section 8.3, we explore working patterns within the high-level event log, using frequent pattern mining to identify common working patterns among resources.

8.1 Forming High-Level Event Log

The primary challenge addressed in this section is the formation of a high-level event log, which is a crucial step before analyzing resource working behavior and patterns. The high-level event log provides a structured representation of resource activities, capturing the aggregated workflow of re-

sources within Company C's sterilization process.

In this thesis research, the high-level event log (HL) is constructed by building a sequence of high-level batches, or high-level traces, for each resource r and each day d. Specifically, for every resource r on a given day d, we construct a high-level trace $\gamma(r, d)$ as follows:

$$\gamma(r, d) = \langle h_1, h_2, \ldots, h_n \rangle$$

where each h_i represents a high-level event (or batch) within the trace. These high-level events are derived from the high-level batch aggregation process discussed in Chapter 7.

The construction high-level event log represents the collection of each trace $\gamma(r, d)$ in the data. The process of high-level event log construction involves the following key steps:

1. Defining High-Level Events

   The first step in the methodology is defining high-level events. As established in the previous chapter, we perform high-level batch aggregation to group related BatchInstances into cohesive high-level batches. Each high-level batch is treated as a distinct high-level event. A high-level event is characterized by the following attributes:

   (a) The specific activities it includes.

   (b) The resource responsible for these activities.

   (c) The time frame during which these activities occur.

   Formally, we can define a high-level event hle_i as follows:

   Definition 19. ***(High-Level Event)***

   $$hle_i = \langle \{a_1, a_2, \ldots, a_n\}, r, [t_{start}, t_{end}] \rangle$$

   where:

   - $\{a_1, a_2, \ldots, a_n\}$ *represents the set of activities included in the high-level event.*
   - r *represents the resource responsible for the high-level event.*
   - $[t_{start}, t_{end}]$ *represents the time frame during which the activities occurred.*

   This formal definition ensures that each high-level event is uniquely identified by its activities, the resource involved, and the time period over which the event spans.

2. Extracting Data from the Event Knowledge Graph (EKG)

   With the high-level events defined, the next step involves extracting the necessary data from the Event Knowledge Graph (EKG). We utilize Neo4j's Cypher query language to retrieve relevant data, including:

   (a) Timestamps, activities, resource IDs of the high-level events.

   (b) Additional attributes such as batch numbers included, the number of kits processed, number of low level events, and whether the high-level event involved collaboration with other resources.

   This extracted data is then processed into a structured format, ensuring that all dates, timestamps, and other attributes are correctly formatted and ready for further processing.

3. Creating the Complete High-Level Event Log

   The final step is constructing the high-level event log. The high-level events are sequenced chronologically based on their date and earliest timestamp. This ordered sequence reflects the resources' workflow over a day, providing a clear timeline of activities.

By following these steps, we create a high-level event log that captures resource workflows within the sterilization process. This log is now ready for further analysis, allowing to explore resource behavior and identify patterns.

8.2 Resource Working Behavior Based on High-Level Batching and Workload

In this subsection, we focus to a detailed examination of resource working behavior within Company C's sterilization process. Our hypothesis, derived from multiple iterations of analysis over sterilization process, batching, and high-level batching, posits that resource behavior is significantly influenced by workload in the sterilization center. Essentially, the hypothesis suggests that resources are more likely to be directed towards areas where work is accumulating. Through this analysis, we seek to answer a key question: How do resources respond to increasing workloads at different stations?

To validate this hypothesis and gain understanding of resource working behavior, we first define the workload and perform an in-depth analysis that will identify the correlation between workload and the corresponding actions taken by resources on the high-level aggregation.

Understanding how resources work are important for achieving the main objective of this thesis provide high-level resource features for development of Auto-Twin digital twin for company C that accurately mirrors real-world operations.

8.2.1 Defining the Workload

Before explaining the approach used for analyzing resource behavior, it is important to define the concept of workload in the context of Company C's sterilization process. Workload, in this setting, refers to the amount of work accumulated at various activities (process stages), specifically the number of kits waiting to be processed on process stages within a specific time frame.

The formal definition of the workload, along with the detailed method for calculating it and analyzing its correlation with high-level resource activities, will be outlined in the following Subsection 8.2.2.

8.2.2 Detailed Approach for Analyzing Resource Working Behavior

To conduct an analysis of resource working behavior, we established an approach that utilizes the high-level event log, as defined in Section 8.1, alongside the calculated workload, represented by the number of kits awaiting for processing. This approach involves time-based analysis, enabling us to track the dynamics between workload and the corresponding resource behavior.

We utilize a 30-minute time bin in our analysis because it provides an effective balance between capturing detailed insights into resource behavior and keeping the data manageable. This interval is sufficient to observe significant shifts in workload and resource allocation, while still being short enough to accurately reflect changes in the process.

We now proceed to the detailed steps of the approach, which are outlined as follows:

1. **Calculation of the Workload Utilizing Buffer and Start Kit Values**: The first step in our approach involved determining the workload by calculating the buffer and start values.

   To perform this buffer and start calculation, we utilized the event log provided by Company C. This log, after undergoing a thorough cleaning process, was loaded into the EKG, details of which provided in the Chapter 5. The initial event log allows to track the flow of kits through the sterilization process, enabling monitoring the buffers at each activity (station). By analyzing these flows and transitions, we were able to

accurately measure the buffers at each station within 30-minutes time bins, thereby quantifying the workload.

The following steps will detail the specific procedures used to calculate these buffers.

(a) **Initial Event Log Preprocessing:**

Grouping Events and Sorting within Traces: The first step in the preprocessing involved organizing the initial event log data (L) by grouping low-level events according to their `runId`. The `runId` serves as the case identifier for kit traces within the event log, with each `runId` representing a unique trace corresponding to a specific kit as it progresses through the sterilization process.

Definition 20. ***(Trace)*** *Each trace σ is defined as a sequence of events:*

$$\sigma = \langle e_1, e_2, \ldots, e_n \rangle,$$
$$\textit{where } (runId_x, activity_x, timestamp_x) \in e_x$$

In this context, L denotes the set of all traces, A the set of all activities, and E the set of all events. Grouping by `runId` ensures that we capture the entire sequence of events for each kit.

Subsequently, these events were sorted by `timestamp` to establish a chronological sequence within each trace. This sorting is required for tracking the flow of each kit over time through the sterilization process and for the subsequent calculation of buffers.

Time Binning: Following the grouping and sorting of events, the preprocessed event log was divided into 30-minute time bins.

(b) **Processed Kits Evaluation:**

We continued with an assessment of kits processed on each process activity within each time bin.

Definition 21. ***(Processed Kits)*** *For each activity A in a time bin T_i, the value is equal to the number of kits for which there were performed events (e) with activity (`activity(e)`) following the event log traces (σ):*

$$ProcessedKits(a, T_i) = \sum_{\sigma \in L} |\{e | e \in \sigma \wedge activity\,(e) = a \wedge timestamp(e) \in T_i\}| \;\; \forall a \in A, T_i \in T$$

This calculation is necessary to track the number of kits processed at each station, as it allows us to definitively know what was processed at each activity and with what timestamp based on the initial event log. This step serves as a prerequisite for determining the buffer at each activity.

(c) **Initializing Starting Traces Kits and Entering Buffer Kits**

Start Values To ensure the kits tracking that initiate their processing in each time bin, we calculated the start values.

Definition 22. ***(Start Value)*** *The start value represents the number of kits that begin their trace with a specific activity within a given time bin. For each trace (σ), if the first event (e_1) in the trace corresponds to a particular activity (a) and falls within the time bin (T_i), the start value for that activity and time bin is incremented. This is formally defined as:*

$$Start(a, T_i) = \sum_{\sigma \in L} |\{e_1 | e_1 = \sigma[1] \wedge activity\,(e_1) = a \\ \wedge\, timestamp(e_1) \in T_i\}| \;\forall a \in A, T_i \in T$$

This calculation captured the initiation of processes, providing insight into how and where kits start their process flows.

Entered Buffer Values In addition to start values, we calculated entered buffer values to identify kits that entered the buffer during each time bin and awaited processing in subsequent events.

Definition 23. ***(Entered Buffer Kit Value)*** *The entered buffer kit value for a specific activity and time bin reflects the accumulation of kits from previous events in the trace. For an event (e_j) in a trace (σ), we look at the preceding event (e_{j-1}) to determine its time bin (T_i) and increment the entered buffer value accordingly. The entered buffer value is formally defined as:*

$$EnteredBuffer(a, T_i) = \sum_{\sigma \in L} |\{e_j | 1 < j \leq |\sigma| \wedge e_{j-1} = \sigma[j-1] \\ \wedge\, e_j = \sigma[j] \wedge activity(e_j) = a \\ \wedge\, timestamp(e_{j-1}) \in T_i\}| \;\forall a \in A, T_i \in T$$

Entered buffers were required to evaluate when kit was placed to wait for processing.

Addressing the First Event in Traces It is important to note that the first event of a kit's trace is not included in the *EnteredBuffer* values for a specific activity, as it lacks a preceding event in the trace and we do not know when kit was placed for awaiting for sterilization on its' initial trace event. However, the first event is still accounted for in the *ProcessedKits* for that activity. This distinction could lead to inaccuracies in subsequent buffer calculations. To mitigate this, we introduced the *Start* value, which captures events that initiate a trace, ensuring tracking of kit flows from the beginning of their processing.

(d) **Buffers Calculation**

The buffer values for each activity a and time bin T_i were calculated to account for kits remaining in the buffer from previous time bins. This calculation ensure that we tracked the flow of kits through the sterilization process, accounting for those that were not processed in earlier time bins:

Definition 24. ***(Buffer)*** *We define buffer as follows:*

$$Buffer(a, T_i) = \begin{cases} EnteredBuffer(a, T_i) & \text{if } i = 1 \\ \\ EnteredBuffer(a, T_i) + Start(a, T_{i-1}) & \\ +Buffer(a, T_{i-1}) - ProcessedKits(a, T_{i-1}) & \text{otherwise} \end{cases}$$

We only updated buffer values for $i > 1$ as the first bin value was set correctly during the initialization phase. This step is important for understanding the carryover of unprocessed kits, which can affect subsequent time bins' workload.

(e) **Workload Calculation** The workload for each activity a and time bin T_i was calculated by summing the buffer values with the start values for the same activity and time bin. This calculation provided a comprehensive measure of the total number of kits awaiting processing within the sterilization process.

Definition 25. ***(Workload)*** *We define workload as follows:*

$$Workload(a, T_i) = Buffer(a, T_i) + Start(a, T_i)$$

For the activity *Entrada Material Sucio*, we only have start values and no buffer values, as it represents the initial stage of the process where kits first enter the sterilization.

2. **Prepartion for Correlation Analysis**:

   (a) **Normalization of Workload Values** To perform the correlation analysis, it was required first to normalize workload values for each time bin to ensure comparability. The workload, which represent the accumulated kits waiting for processing, were computed based on the previous step. Normalization was then applied to the workload values to ensure that the data was on a consistent scale for performing correlation analysis .

   (b) **Preparation of the High-Level Event Log:** The high-level event log (*HL*), which was established in Section 8.1, was further processed to prepare it for correlation analysis. The steps included:

      i. Aggregating the data to represent the number of kits processed for each activity within each high-level batch.

      ii. Time-binning the aggregated data, similar to the approach used for the workload values calculation. However, for the high-level event log we considered earliest timestamp attribute to be able to associate high-level event with a particular time bin.

      iii. Normalizing the data across activities to remove any scale effects and ensure that the correlation analysis could capture the relationships between workload and resource behavior.

3. **Performing Correlation Analysis Between workload Values and Prepared High-Level Event Log:** Once the workload values and high-level event log were prepared and normalized, we conducted a correlation analysis to examine the relationship between the workload and the resource workflow (captured in the high-level event log). This analysis aimed to identify how variations in workload influenced performing work over kits on the process activities and transitions observed in the high-level event log.

4. **Results Evaluation:** The results of the correlation analysis were evaluated to determine the significance and patterns of the relationships between the workload values and high-level events. These results are

detailed in the next subsection, where we present specific findings and their implications for understanding resource behavior.

In conclusion, the detailed approach outlined in this subsection provides a methodology for analyzing resource working behavior by utilizing the high-level event log and calculated workload.

8.2.3 Results Evaluation of Analyzing Resource Working Behavior

After explaining the approach used to analyze resource behavior, we now present the results. The main goal of this analysis is to test our hypothesis that the resource work in Company C's sterilization process is influenced by the number of kits waiting to be processed at different stations. We aim to understand how the accumulation of kits impacts resource allocation. To do this, we conducted a correlation analysis using data from consistent time bins, focusing on the relationship between workload and the number of kits processed in high-level batches at different activities.

The correlation matrix in Figure 8.1 was created using these consistent time bins to maintain accuracy in our analysis. The color gradient shows the strength and direction of the correlations, with dark red indicating strong positive correlations and dark blue indicating strong negative correlations.

This correlation matrix helps us understand how workload on specific activities influenced the work performed on those activities. By analyzing the relationships between the workload values (named as Workload_activity name) and the processed kits values from the high-level event log (named as kits_activity name), we can see which workload levels at certain activities impacted the actual work done.

To evaluate the hypothesis that the behavior of resources within Company C's sterilization process is influenced by the accumulation of kits at various stations awaiting sterilization, we must analyze the correlation matrix with a focus on understanding how workload (buffer and start values) at each station affects the number of kits processed. This analysis allows us to determine whether the workload indeed influences resource behavior and processing efficiency across different stages of the sterilization process.

Entrada de Material Sucio (Initial Process Stage): The "Entrada de Material Sucio" is the first stage in the process, and as such, there are no workload values available at this point since kits have not yet accumulated before processing begins. This is why we only observe the start value in the correlation matrix. The correlation between "Start Entrada Material Sucio"

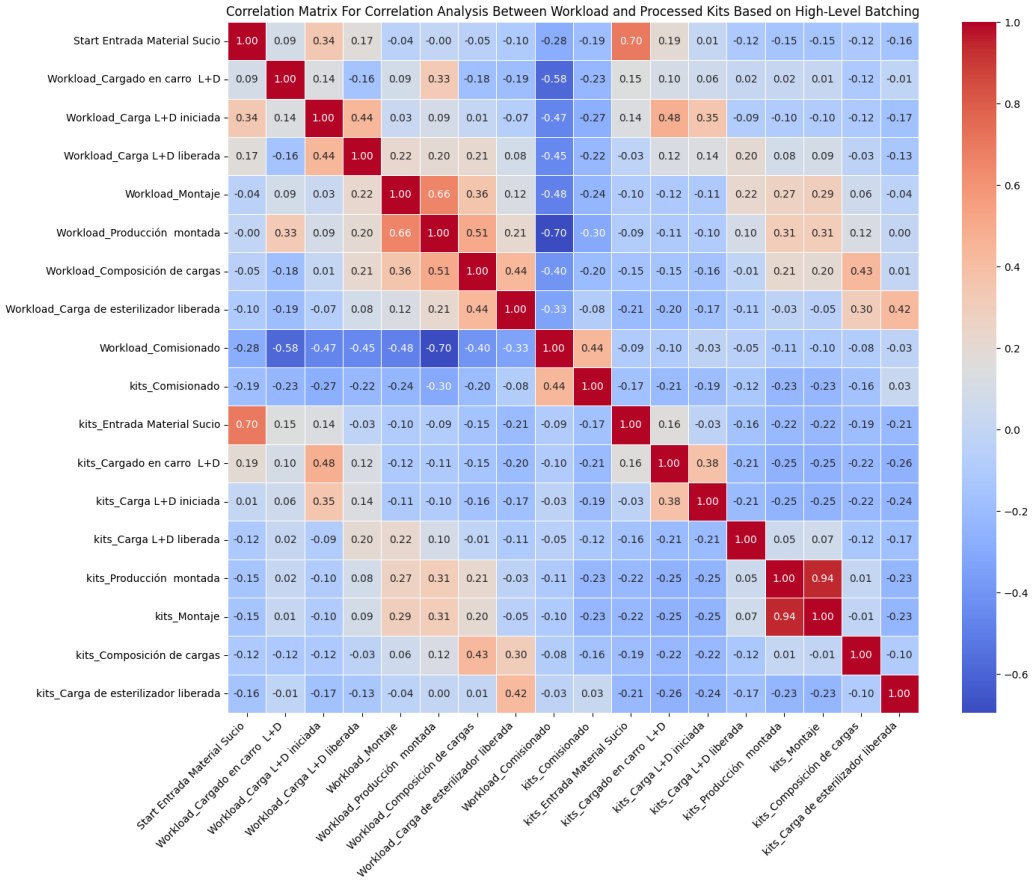

Figure 8.1: Correlation Matrix for Correlation Analysis Between Workload Values and Processed Kits Based on High-Level Batching Aggregation

and "kits_Entrada Material Sucio" is high at 0.70, indicating that the initial inflow of kits directly influences the workload at this stage. This suggests a demand-driven process, where the number of incoming kits closely aligns with the number of kits processed, as expected.

Additionally, there is a moderate positive correlation of 0.34 between "Start Entrada Material Sucio" and "Workload_Carga L+D iniciada". This suggests that the start of the initial process stage not only affects the immediate workload of processing kits but also has some influence on the subsequent stages, particularly where the kits are being prepared for the next steps ("Carga L+D iniciada"). This indicates that the initial kit inflow could partially predict the workload in subsequent processing stages, demonstrating the interconnected nature of the workload across different stages of the process.

Cargado en carro L+D: The correlation between "Workload_Cargado en carro L+D" and "kits_Cargado en carro L+D" is relatively low at 0.19. This suggests that the number of kits loaded onto the washing rack is not as tightly linked to the workload as one might expect. One possible reason for this weaker correlation could be the limited capacity of the washing machines in the sterilization center, which might restrict the number of kits that can be loaded simultaneously, leading to a less direct relationship between the number of kits and the associated workload.

Interestingly, the workload at this stage has a moderate positive impact on "Workload_Producción montada" with a correlation of 0.33. This indicates that the workload involved in loading kits onto the washing rack does influence the workload in the subsequent production stage, specifically during the assembly phase ("Montaje"). This connection suggests that while the initial loading might be constrained by machine capacity, it still plays a significant role in preparing for and sustaining the workload in later stages of the process.

Carga L+D iniciada: There is a moderate correlation of 0.34 with Start "Entrada Material Sucio", indicating that as initial materials enter, the workload at this stage increases. This shows how the start of the process impacts subsequent stages. A correlation of 0.44 with "Workload_Carga L+D liberada" reflects a strong connection between the initiation and completion of the load and distribution process, suggesting a steady flow through this stage.

In addition to the previous, "Workload_Carga L+D Iniciada" correlates 0.48 with "kits_Cargado en carro L+D", indicating that as more kits are loaded onto the washing rack, the workload at this stage rises. This shows a direct relationship between kit preparation and the workload required to start processing. Finally, a 0.35 correlation with "kits_Carga L+D iniciada" itself underscores the link between the number of kits being processed and the workload at this initiation stage.

In summary, "Workload_Carga L+D Iniciada" is closely tied to the initial material inflow, the progression of workload, and the preparation of kits, making it one of the important point in the workflow.

Carga L+D liberada: There is a moderate correlation of 0.44 between "Workload_Carga L+D Liberada" and "Workload_Carga L+D Iniciada". This suggests a strong linkage between the initiation and completion of the load and distribution tasks, indicating that once the process begins, it progresses steadily towards completion. This steady progression likely reflects the con-

tinuous operation of the washing machines, ensuring that once a load is started, it is processed without significant delays.

Additionally, there is a weaker correlation of 0.20 between "Workload_Carga L+D Liberada" and "kits_Carga L+D liberada". This indicates that while there is some relationship between the workload and the number of kits being released, it is not as strong as the correlation with the workload initiation. This could be due to the capacity limitations of the washing machines, where the number of kits processed might not directly scale with the workload due to the fixed capacity of each machine.

In summary, "Workload_Carga L+D Liberada" is closely tied to the progression of the workload through the load and distribution stages, with a steady flow from initiation to completion. However, the relationship with the number of kits processed is less direct.

Montaje: The "Montaje" stage, which focuses on the beginning of assembly process, has significant correlations between its workload "Workload_Montaje" with several other stages. A strong correlation of 0.66 between "Workload_Montaje" and "Workload_Producción montada" suggests that as the workload in assembly increases, it directly impacts production. where "Producción montada" is the finalization of assembly. This indicates that higher demands in assembly can lead to increased workloads in finished assembly, highlighting the interdependence between these stages.

Additionally, "Workload_Montaje" shows a moderate correlation of 0.36 with "Workload_Composición de cargas", indicating that the assembly workload rises alongside the workload associated with load of kits into sterilization maschines.

The assembly workload ("Montaje") also affects the handling of kits, with a 0.30 correlation to "kits_Montaje" and 0.26 to "kits_Producción montada". These correlations indicate that as the workload in assembly increases, it often leads to a higher number of kits being processed in both the assembly and subsequent production stages. A weaker correlation of 0.22 with "kits_Carga de esterilizador liberada" suggests that an increase in assembly workload can slightly impact the processed kits number at the sterilization release stage (where kits taken from machines after sterilization). However this effect is less significant.

In summary, the workload on "Montaje" stage significantly impacts both the process kits on "Montaje", finalized assembly on "Producción montada" and the progression of kits through sterilization stage.

Producción Montada: The "Workload_Producción Montada" stage, which represents the finalization of the assembly, has notable correlations with various stages in the workflow. A strong correlation of 0.66 with "Workload_Montaje" indicates that as the workload in assembly increases, the demands at the production finalization stage also rise. This close relationship shows that the beginning of assembly process directly impacts the workload required to complete the assembly.

Additionally, "Workload_Producción Montada" has a moderate correlation of 0.51 with "Workload_Composición de Cargas", suggesting that as the workload in the finalizing the assembly increases, the workload on the stage of leading kits into sterilization machines experiences a corresponding increase in workload. A 0.33 correlation with "Workload_Cargado en Carro" further indicates that the workload on initial loading of kits onto the washing rack has an impact on the workload in the final stages of assembly in a smaller extend.

The stage has an impact the processing of kits, as shown by the 0.31 correlation with "kits_Montaje" and 0.30 with "kits_Producción montada". These correlations suggest that as the workload in assembly and finalizing assembly increase, it leads to a greater number of kits being processed in both stages. This underscores the interconnectedness of these stages.

Composición de Cargas: The "Workload_Composición de Cargas" is related to the workload on the stage on which kits are placed into sterilization machines. The workload on this stage is significantly impacted by the workloads at preceding stations, particularly "Workload_Montaje" and "Workload_Producción Montada", as previously mentioned.

In addition, there is a moderate correlation detected of 0.44 between "Workload_Composición de Cargas" and "Workload_Carga Esterilizador Liberada", indicating that the workload at this stage influences the workload at the subsequent stage where machines are freed after sterilization. This suggests that as the workload in preparing and loading kits for sterilization increases, it directly impacts the workload on the releasing kits from the sterilization machines process stage.

Moreover, the "Workload_Composición de Cargas" stage shows a 0.43 correlation with the number of processed kits at this stage. This correlation indicates that the workload here not only prepares kits for sterilization but also significantly impacts the actual processing of these kits.

Carga de esterilizador liberada: The workload on "Carga de Esterilizador Liberada" stage is closely influenced by the number of kits processed

on the preceding "Composición de Cargas stage". The correlation of 0.30 between the kits processed at "Composición de Cargas" and the workload at "Carga de Esterilizador Liberada" suggests that the number of kits prepared in the previous stage directly impacts the workload in freeing up the sterilization machines.

Additionally, there is a 0.42 correlation between "Workload_Carga de Esterilizador Liberada" and the number of processed kits at this stage.

Comisionado(Final Process Stage): The workload on "Comissionado" stage shows a notable negative correlation with workloads and processed kits from the preceding process stages. Specifically, it has a strong negative correlation of -0.58 with "Workload_Cargado en Carro" and an even stronger negative correlation of -0.70 with "Workload_Producción Montada". These negative correlations suggest that as the workloads increase in the earlier stages, the workload in "Comissionado" tends to decrease. This could indicate that when earlier stages are busier, there is less capacity or need for workload at the "Comissionado" stage, or it could reflect a shift in focus as the process advances.

Despite these negative correlations, the workload at "Comissionado" does have a positive impact on the number of processed kits within this stage, with a correlation of 0.44. This suggests that as the workload at "Comissionado" increases, it leads to a higher number of kits being successfully processed at this stage

8.2.4 General Overview of Resource Working Behavior

The analysis reveals that the work in Company C's sterilization process is significantly influenced by the number of kits awaiting processing at various stages. This relationship is evident in several key stages of the process. For instance, at the "Entrada de Material Sucio" stage, the workload, which consists of kits waiting to enter the process, directly influences the number of kits processed at this initial stage. Similarly, the workload at "Montaje", "Carga L+D Iniciada", "Producción Montada", "Composición de Cargas", "Carga de Esterilizador Liberada", "Comisionado" impacts the number of kits processed on this stations.

Another important finding is the relationships between workloads at different stations. The strong correlation between "Workload_Montaje" and "Workload_Producción Montada" highlights the interdependence between these stages, where an increase in workload at the assembly stage directly impacts the completion of the assembly process. Additionally, the workload at "Composición de Cargas" significantly influences the workload in subsequent

stages, such as "Carga de Esterilizador Liberada", showing that the accumulated kits in earlier stages can directly impact the capacity and efficiency of processing kits in later stages.

Notably, the "Comisionado" stage shows significant negative correlations with the workloads and processed kits from preceding stages, particularly with "Workload_Cargado en Carro" and "Workload_Producción Montada". This suggests that as the workloads in earlier stages increase, the workload in "Comisionado" tends to decrease. Despite these negative correlations, the workload at "Comisionado" positively impacts the number of kits processed at this final stage, indicating that even with a reduced workload, this stage is critical for the final processing of kits.

Overall, these findings underscore the complex interplay between different stages of the process, where the accumulation of kits at each stage has both direct and indirect effects on the number of kits processed and the overall efficiency of Company C's sterilization process.

8.3 Frequent Working Patterns Using High-Level Aggregation

The second part of the high-level resource behavioral analysis is identifying common working patterns. Our goal is finding frequent common working patterns among users working on a particular day of the week and in a particular shift. This part of the analysis similarly to the analysis of resource working behavior presented in the Section 8.2 provides insides into resource working behaviour and contribute to the development of a digital twin for Company C.

We start this section with establishing a methodology that was used for this part of the analysis of common working patters in the Subsection 8.3.1 and finalise with the results evaluation in the Subsection 8.3.2

8.3.1 Detailed Approach for Analyzing Frequent Working Patterns

To identify frequent working patterns, we employed the following methodology:

1. **Event Log Enrichment:** Utilizing the high-level event log established in Section 8.1, we enriched the data by incorporating resource shift details. Each high-level event was assigned a shift value based on a day and a specific shift during which a resource was active. The description

of shifts presented in the Section 2.3. This enrichment ensured that each event was associated to the corresponding shift. Additionally, we extended the high-level event log with a week day parameter in order to find common patterns on particular day of the week and shift.

2. **Frequent Sequence Mining:** For each resource, we considered the sequences of high-level events based on the specific date, weekday, and shift. Among all sequences that occurred on the same shift and weekday, we focused on identifying common patterns, whether they were individual high-level events or sequences of such events. To identify these, we employed the FP-Growth algorithm, that enabled identification of the most prevalent sequences of high-level events reflecting typical behaviors during specific weekdays and shifts.

3. **Analysis and Interpretation:** The identified sequences were analyzed to assess their significance and prevalence. The analysis focused on the following key metrics:

   (a) **Traces with Pattern:** The number of high-level batch traces for resources on a particular day of the week and shift where the identified sequence was present.

   (b) **Traces on Weekday and Shift:** The total number of high-level batch traces for resources on a particular day of the week and shift.

   (c) **Percentage:** The proportion of traces containing the pattern relative to the total number of traces, indicating the significance of the identified sequence. For the analysis percentage values lower than 15% were not considered, as they are not significant given the relatively low number of traces on each weekday and shift.

8.3.2 Results Evaluation of Frequent Working Patterns Analysis

In this subsection, we provide a discussion of the results obtained from our analysis, as presented in the Appendix A. The analysis of the frequent working patterns across different shifts and weekdays revealed several key insights:

1. **General Patterns:**

   (a) The most frequent patterns often consist of a single high-level batch, predominantly involving one type of activity. This indicates

a tendency for resources to perform repetitive tasks within a day or shift.

(b) The pattern "Carga L+D liberada (Separate)" appeared frequently across multiple shifts, particularly in Shifts 2 and 4, indicating its critical role in the workflow. Similarly, "Producción montada, Montaje (Together)" emerged as a common collaborative pattern in Shift 1, particularly on Monday, Tuesday, and Wednesday.

2. **Shift 1:**

(a) On **Monday**, the primary patterns were "Carga L+D iniciada, Cargado en carro L+D (Together)" with a 19.64% occurrence, "Montaje (Together)" at 16.07%, and "Producción montada, Montaje (Together)" at 26.79%. These patterns suggest that Shift 1 on Monday is characterized by collaborative work involving multiple resources.

(b) **Tuesday** displayed similar trends, with "Montaje (Together)" occurring in 36.21% of traces, highlighting the collaborative nature of Shift 1 during this day. The pattern "Carga de esterilizador liberada (Separate)" was also significant, with 34.48% occurrence, indicating a focus on releasing sterilizers.

(c) On **Wednesday**, the patterns "Montaje (Together)" and "Producción montada, Montaje (Together)" continued to dominate, with 30.16% and 28.57% occurrence, respectively. This suggests that collaborative activities are consistently important in Shift 1 throughout the week.

(d) **Thursday** and **Friday** also reflected a high occurrence of collaborative patterns, with "Producción montada, Montaje (Together)" reaching 29.09% on Thursday and 38.00% on Friday, emphasizing the teamwork involved in these tasks.

3. **Shift 2:**

(a) On **Monday**, the pattern "Carga L+D liberada (Separate)" was prevalent in 29.41% of traces, reflecting its importance in this shift. Other frequent patterns included "Carga de esterilizador liberada (Separate)" at 23.53% and "Composición de cargas (Separate)" at 17.65%.

(b) **Tuesday** and **Wednesday** continued this trend with "Carga L+D liberada (Separate)" appearing in 31.25% and 41.18% of traces,

respectively. This indicates that Shift 2 consistently involves crucial individual tasks related to the handling and releasing of loads.

(c) On **Thursday**, the same pattern was observed with "Carga L+D liberada (Separate)" appearing in 30.00% of the traces, reinforcing its significance during Shift 2 across multiple days.

4. **Shift 3:**

   (a) Shift 3 exhibited lower pattern percentages, indicating a more diverse range of activities with less repetition. For example, on **Tuesday**, "Comisionado (Separate)" occurred in 17.65% of traces, and on **Friday**, "Producción montada, Montaje, Composición de cargas (Together)" was present in 17.39% of traces. This suggests that Shift 3 may involve a wider variety of tasks, leading to less frequent repetition of specific patterns.

5. **Shift 4:**

   (a) On **Monday**, the pattern "Entrada Material Sucio (Separate)" was present in 32.00% of the traces, indicating its critical role during this shift. This pattern continued to be significant on **Wednesday** with a 41.67% occurrence and on **Thursday** with 35.56%.

   (b) **Friday** also reflected the importance of "Entrada Material Sucio (Separate)" with a 32.61% occurrence. The consistent presence of this pattern during Shift 4 across multiple days highlights its importance in the workflow, particularly in handling entering the sterilization kits.

6. **Saturday and Sunday:**

   (a) **Saturday** Shift 1 showed a 29% occurrence of the pattern "Carga L+D iniciada, Cargado en carro L+D (Separate)", indicating the routine nature of this task on weekends. Shift 4 on Saturday had a higher occurrence of "Carga L+D iniciada, Cargado en carro L+D (Separate)" at 54% performed by one resource.

   (b) On **Sunday**, Shift 1 patterns included "Carga de esterilizador liberada (Separate)" and "Composición de cargas (Separate)" each at 31%. Shift 4 showed the same pattern with a 50% occurrence, indicating that material handling and load initiation are crucial on Sundays and performed by one resource.

Insights Regarding Collaboration within Shifts

- **Shift 1:** This shift often includes combined activities like "Producción montada, Montaje," reflecting a more collaborative approach. The frequent occurrence of such patterns highlights the teamwork involved in completing these high-level events, particularly on weekdays.

- **Shifts 2 and 4:** These shifts are characterized by a higher prevalence of separate activities, particularly "Carga L+D liberada" and "Entrada Material Sucio". These shifts involve significant individual high-level events handling, reflecting the specialized nature of the tasks performed.

- **Shift 3:** Shows a mix of both separate and combined high-level events sequences, with the smallest values of pattern percentages, indicating high diversity in high-level working behavior. This suggests that Shift 3 may involve more varied tasks, with resources handling different types of activities, leading to less repetition of specific patterns.

General Overview of Work Patterns and Expectations

The analysis of work patterns across different shifts and weekdays reveals distinct operational behaviors that align with the specific demands of each shift.

Shift 1 is predominantly characterized by collaborative activities, especially on weekdays such as Monday, Tuesday, and Wednesday. In approximately 30% to 40% of the traces, resources are engaged in joint tasks, such as "Producción montada, Montaje (Together)." This high level of collaboration underscores the importance of teamwork during Shift 1.

In contrast, **Shifts 2** and **4** are more focused on individual task handling. Patterns such as "Carga L+D liberada (Separate)" and "Entrada Material Sucio (Separate)" frequently appear, accounting for about 20% to 35% of the traces. These shifts play a crucial role in maintaining the steady flow of operations by consistently performing routine tasks that require a high degree of autonomy and expertise. The repetition of these patterns across multiple days highlights their importance in the overall workflow.

Shift 3 experiences the most diverse work patterns, with no single sequence dominating more than 20% of the traces. This diversity suggests that resources in Shift 3 are likely to encounter a broader range of tasks, leading to a dynamic and varied work environment.

Overall, the findings from this analysis strongly support the thesis of High-Level Resource Behavioral Analysis. The results demonstrate that the

nature of work in each shift is closely aligned with the specific operational needs of those periods. Shift 1 is characterized by collaborative efforts, while Shifts 2 and 4 are focused on specialized, individual tasks. Shift 3 introduces flexibility into the workflow by accommodating a wide range of activities.

These insights not only improve the understanding of how work is distributed across different shifts but also provide an overview of the frequent patterns within shifts and weekdays. This is important for the development of a digital twin of the sterilization process in Company C, as it offers a solid foundation for accurately modeling resource behavior in a dynamic operational environment.

Chapter 9

Conclusion

This thesis presents an analysis of resource behavior and working patterns within the sterilization process at Company C, utilizing process mining techniques to uncover deep insights into resource operational patterns and task prioritization.

This thesis began with an initial exploration of data, highlighting batching processes and shift-based work patterns within Company C's sterilization process. The research initially aimed to apply the Task Instance Framework, using Company C' EKG as part of the AUTO-Twin project, for detailed behavioral analysis. However, this framework was found incompatible with the specific needs of the dataset, leading to the development of alternative batch detection techniques.

Two methodologies, Batching over Resource and Batching over Activity, were introduced. Batching over Activity was chosen for its effectiveness in defining collaborative work and providing a more streamlined analytical framework. Despite its benefits, this approach alone did not address the repetitive work patterns among resources, leading to the adoption of high-level aggregation techniques. These techniques categorized work patterns as regular, iterative, or complex iterative, facilitating structured analysis of resource behavior.

The research utilized a high-level aggregation log, formed based on high-level batches of the EKG , to examine the impact of workload—defined as kits awaiting processing—on the processed kits on the sterilization stages comprehensively.

For this research we provide the code presented in the GitHub repository that reflects the implementation of this thesis steps.

It was found that workload not only directly influences the processing of kits but also impacts subsequent stages, illustrating the interconnected nature of the process workflow. Additionally, frequent sequence mining was

---

GitHub Repository: https://github.com/Liliia-Aliakberova/ThesisResearchHighLevelBatchAggregation

employed to identify common work patterns across different weekdays, and shifts, uncovering distinct collaborative behaviors and operational nuances within shift structures.

The findings from this study provide valuable insights into the dynamics of resource allocation and task execution in complex operational settings, laying the groundwork for the development of effective digital twins.

In summary, this thesis not only explores the complexities of analyzing resource behavior within a sterilization context but also provides the way for future advancements in process mining and providing high-level resource features for digital twin implementation.

9.1 Limitations

While the study achieved its objectives, several limitations should be acknowledged. One of the main limitations is the reliance on correlation analysis to infer the relationships between workloads and processed kits. Although correlations provide valuable insights, they do not establish causality, and further studies are required to explore these relationships more deeply, possibly through causal analysis or experimental design.

Another limitation lies in the scope of high-level aggregation. The current approach limited the depth to three nodes to avoid overgeneralization. However, this restriction may have prevented the capture of more complex iterative behaviors that could be critical for a complete understanding of the process. Future research should consider ways to extend the depth of aggregation without compromising the clarity and precision of the analysis.

The approach to batch detection in this thesis primarily focused on specific activities. While this provided detailed insights into those stages, it did not fully account for how batches are formed across multiple activities, particularly those exhibiting iterative behavior. Expanding the detection method to consider multiple activities and refining time constraints could yield a more comprehensive understanding of batching in the process.

Lastly, the analysis of resource behavior focused on identifying patterns within individual and group activities, but it did not fully explore how resources collaborate or how their actions during shifts impact overall process efficiency. For the digital twin to more accurately replicate real-world conditions, future research should investigate collaboration patterns among resources to understand how the actions of one resource influence others. Additionally, studying resource behavior during shifts, including the overlaps between shifts, could provide valuable insights into resource allocation and transitions, further enhancing the accuracy and effectiveness of the sim-

ulation model.

9.2 Future Work

This thesis has focused on understanding human resource behavior and working patterns in the sterilization process at Company C, with the aim of providing high-level features for the development of a digital twin. While significant progress has been made, there are several areas where further research could enhance the findings, help to solve limitations mentioned in the previous Section 9.1 and contribute to a more accurate and effective digital twin model. Below are some potential directions for future work.

Batch Detection Techniques:

1. **Exploring Different Batching Detection Methods:** Future research could investigate various methods for detecting batches by analyzing the arrival times of kits at different stations. By adapting time constraints for batching to each activity individually, the detection process could become more precise and better aligned with the specific characteristics of each stage.

2. **Defining Batching Across Multiple Activities:** Another important area for research is defining batching not only within individual activities but also across multiple activities, particularly those that frequently exhibit iterative behavior. This would offer a more comprehensive understanding of how batches and resources with these batches move through the entire sterilization process.

High-Level Batching Aggregation

1. **Extending the Depth of High-Level Aggregation:** The current analysis focused on complex iterative behaviors within a depth of three nodes. Extending this depth could potentially involve more than 30% of the process, considering that the sterilization workflow comprises nine distinct stages. Future research should explore how to extend the depth of high-level aggregation while ensuring it does not exceed 30-40% of the total process, to maintain analytical clarity and precision.

2. **Developing New Aggregation Methods:** Future work could explore creating more compact methods for high-level aggregation that effectively capture complex behaviors while maintaining a clear focus

in the analysis. The approach could involve fully abstracting from time to focus solely on the sequence of resource activities. Instead of creating separate high-level nodes for each BatchInstance patterns where a resource performs a batch activity at different times, we could represent each unique batching activity with a single high-level node. The workflow would then be depicted through nodes, with edges capturing the sequence and return of the resource to specific activities. This approach would reduce the space needed for analysis and allow us to concentrate on the sequence of activities, providing a broader view of resource behavior in the process.

Resource Working Behavior Analysis

1. **Identifying Collaboration Patterns Among Resources:** Future research could explore how groups of resources frequently work together. By analyzing the behavior of each resource individually, it could become clearer how the actions of one resource affect others, potentially revealing collaboration patterns that enhance efficiency.

2. **Analyzing Resource Behavior During Shifts:** Further study could focus on how resources behave within shifts, including the overlaps between shifts. This could offer insights into resource allocation and transitions between shifts.

Understanding resource behavior is crucial for developing a digital twin simulation that accurately reflects real-world conditions. The simulation model needs to capture how workers decide whether to stay at a station or move to another. Integrating these decision-making patterns into the digital twin is key to accurately mirroring real-world scenarios and improving the model's effectiveness.

These future research directions could deepen our understanding of batch detection, high-level aggregation, and resource behavior into the sterilization process, providing valuable insights for the development of a more effective digital twin for Company C.

Bibliography

[1] AUTO-TWIN Project. `https://www.auto-twin-project.eu/`. Accessed on on 1th July 2024.

[2] Wil Aalst. *Process Mining: Discovery, Conformance and Enhancement of Business Processes*, volume 136. Springer, 01 2011.

[3] R. Agrawal and R. Srikant. Mining sequential patterns. In *Proceedings of the Eleventh International Conference on Data Engineering*, pages 3–14, 1995.

[4] Reinhard Diestel. *Graph theory*. Springer (print edition); Reinhard Diestel (eBooks), 2024.

[5] Stefan Esser and Dirk Fahland. Multi-dimensional event data in graph databases. *Journal of Data Semantics*, 10:109–141, 2021. Received 29 May 2020, Revised 12 November 2020, Accepted 06 March 2021, Published 27 May 2021, Issue Date June 2021.

[6] Dirk Fahland. Extracting and pre-processing event logs. *CoRR*, abs/2211.04338, 2022.

[7] Dirk Fahland. Multi-dimensional process analysis. In Claudio Di Ciccio, Remco Dijkman, Adela del Río Ortega, and Stefanie Rinderle-Ma, editors, *Business Process Management*, pages 27–33, Cham, 2022. Springer International Publishing.

[8] Dirk Fahland. *Process Mining over Multiple Behavioral Dimensions with Event Knowledge Graphs*, pages 274–319. Springer International Publishing, Cham, 2022.

[9] Nadime Francis, Alastair Green, Paolo Guagliardo, Leonid Libkin, Tobias Lindaaker, Victor Marsault, Stefan Plantikow, Mats Rydberg, Petra Selmer, and Andrés Taylor. Cypher: An evolving query language for property graphs. *Proceedings of the 2018 International Conference on Management of Data*, 2018.

[10] Nicla Frigerio, Baris Tan, and Andrea Matta. Green gateways: a concept for decisions in circular-oriented economies. *Procedia CIRP*, 122:617–622, 01 2024.

[11] Kanika Goel, Tobias Fehrer, Maximilian Röglinger, and Moe T. Wynn. Not here, but there: Human resource allocation patterns. In Chiara Di Francescomarino, Andrea Burattin, Christian Janiesch, and Shazia Sadiq, editors, *Business Process Management*, pages 377–394, Cham, 2023. Springer Nature Switzerland.

[12] Jiawei Han, Jian Pei, and Yiwen Yin. Mining frequent patterns without candidate generation. volume 29, pages 1–12, 06 2000.

[13] Kongzhang Hao, Zhengyi Yang, Longbin Lai, Zhengmin Lai, Xin Jin, and Xuemin Lin. Patmat: A distributed pattern matching engine with cypher. *Proceedings of the 28th ACM International Conference on Information and Knowledge Management*, 2019.

[14] Olaf Hartig. Foundations to query labeled property graphs using sparql. In *SEM4TRA-AMAR@SEMANTiCS*, 2019.

[15] Eva Klijn, Felix Mannhardt, and Dirk Fahland. *Classifying and Detecting Task Executions and Routines in Processes Using Event Graphs*, pages 212–229. CEUR-WS.org, 08 2021.

[16] Eva L. Klijn, Felix Mannhardt, and Dirk Fahland. Aggregating event knowledge graphs for task analysis. In Marco Montali, Arik Senderovich, and Matthias Weidlich, editors, *Process Mining Workshops*, pages 493–505, Cham, 2023. Springer Nature Switzerland.

[17] Orlenys López-Pintado, Marlon Dumas, and Jonas Berx. Discovery, simulation, and optimization of business processes with differentiated resources. *Information Systems*, 120:102289, 2024.

[18] Niels Martin, Marijke Swennen, Benoît Depaire, Mieke Jans, An Caris, and Koen Vanhoof. Batch processing: definition and event log identification. 12 2015.

[19] Gerardo Santillán Martínez, Seppo Sierla, Tommi Karhela, and Valeriy Vyatkin. Automatic generation of a simulation-based digital twin of an industrial process plant. In *IECON 2018 - 44th Annual Conference of the IEEE Industrial Electronics Society*, pages 3084–3089, 2018.

[20] Neo4j. *Cypher Manual*. Accessed on on 16th August 2024.

[21] Boris Otto, Michael Hompel, and Stefan Wrobel. *International Data Spaces: Reference architecture for the digitization of industries*, pages 109–128. 01 2019.

[22] Kornelije Rabuzin and Martina Šestak. Graph database management systems: The past, the present, and the future. In *Encyclopedia of Information Science and Technology, Fifth Edition*, pages 778–790. IGI Global, 2021.

[23] Nick Russell, Wil M. P. van der Aalst, Arthur H. M. ter Hofstede, and David Edmond. Workflow resource patterns: Identification, representation and tool support. In Oscar Pastor and João Falcão e Cunha, editors, *Advanced Information Systems Engineering*, pages 216–232, Berlin, Heidelberg, 2005. Springer Berlin Heidelberg.

[24] Maulshree Singh, Evert Fuenmayor, Eoin P. Hinchy, Yuansong Qiao, Niall Murray, and Declan Devine. Digital twin: Origin to future. *Applied System Innovation*, 4(2), 2021.

[25] Zvonimira Sverko Grdic, Marinela Krstinic Nizic, and Elena Rudan. Circular economy concept in the context of economic development in eu countries. *Sustainability*, 12(7), 2020.

[26] Ava Swevels, Remco Dijkman, and Dirk Fahland. Inferring missing entity identifiers from context using event knowledge graphs. In Chiara Di Francescomarino, Andrea Burattin, Christian Janiesch, and Shazia Sadiq, editors, *Business Process Management*, pages 180–197, Cham, 2023. Springer Nature Switzerland.

[27] Ava Swevels, Dirk Fahland, and Marco Montali. Implementing object-centric event data models in event knowledge graphs. In Johannes De Smedt and Pnina Soffer, editors, *Process Mining Workshops*, pages 431–443, Cham, 2024. Springer Nature Switzerland.

[28] Wil M. P. van der Aalst. *Process-Aware Information Systems: Lessons to Be Learned from Process Mining*, pages 14–17. Springer Berlin Heidelberg, Berlin, Heidelberg, 2009.

[29] Wil M. P. van der Aalst. *Process Mining: Data Science in Action*. Springer, Heidelberg, 2 edition, 2016.

[30] Wil M.P. van der Aalst. *Process Mining in the Large: A Tutorial*, pages 33–76. Springer International Publishing, Cham, 2014.

[31] KW Verhaegh. *Process mining for systems with automated batching.* PhD thesis, Master's thesis, Eindhoven University of Technology, 5612 AZ Eindhoven, 2018.

[32] Yiping Wen, Zhigang Chen, Jianxun Liu, and Jinjun Chen. Mining batch processing workflow models from event logs. *Concurrency and Computation: Practice and Experience*, 25, 09 2013.

[33] Jing Yang, Chun Ouyang, Arthur H. M. ter Hofstede, and Wil M. P. van der Aalst. No time to dice: Learning execution contexts from event logs for resource-oriented process mining. In Claudio Di Ciccio, Remco Dijkman, Adela del Río Ortega, and Stefanie Rinderle-Ma, editors, *Business Process Management*, pages 163–180, Cham, 2022. Springer International Publishing.

List of Figures

2.1 Facility Arrangement of Company C's Sterilization Center . . 16
2.2 Company C Sterilization Process Diagram 17
2.3 Distribution of Events (rows) for Each Resource 22

3.1 Event Knowledge Graph 31

4.1 Thesis Research Methodology Outline 37

5.1 Distribution of Earliest Event Timestamps of Resources 45
5.2 Distribution of Latest Event Timestamps of Resources 46

6.1 Example of Batching over Resource in the Event Knowledge Graph 51
6.2 Example of Batching over Activity in the Event Knowledge Graph 53
6.3 Example of Batching over Resource 54
6.4 Distribution of Events in Batchinstances for Batching over Resource Approach 60
6.5 Example of Resources Return to Previous Activities in EKG . 61
6.6 Iterative BatchInstance Activity Sequences Frequency 62
6.7 An example of concurrent resource utilization 63
6.8 Distribution of Batches where Resources Worked Together within 5-Minute Time Window 64
6.9 Example of Batching over Activity 66
6.10 Illustration of DF_BATCH_RESOURCE edge for "AV" resource 71
6.11 Distribution of Events in BatcheInstances for Batching over Activity Approach 73
6.12 Distribution of BatchInstances Based on the Number of Resources 74
6.13 Distribution of Resources Number per BatchInstance Activity 75
6.14 BatchInstances per Resource and Activity 76
6.15 Iterative Pattern of Batches Performed by Resource "MMF" . 77

6.16 Patterns of Resources Returning to Previous BatchInstances . 78

7.1 Example of Overlapping Batches and Iterative Patterns 81
7.2 Regular Pattern 83
7.3 Iterative Pattern 84
7.4 Complex Iterative Pattern 85
7.5 Singular High-Level Batch 90
7.6 Joint High-Level Batch 91
7.7 Complex Joint High-Level Batch 92
7.8 "AV" Resource High-Level Workflow 93

8.1 Correlation Matrix for Correlation Analysis Between Workload Values and Processed Kits Based on High-Level Batching Aggregation 103

List of Tables

2.1 Number of Events by Process Activity Name and Associated Process Stages 20
2.2 Number of Duplicates per File 21

5.1 EKG Implementation Statistics for Company C 44

6.1 BatchInstances Attributes for Batching over Resource Approach Visualization 55
6.2 Summary of Implementation Results for Batch over Resource Approach 59
6.3 BatchInstance Attributes for Batching over Activity Approach Visualization 65
6.4 Implementation Overview for Batch over Activity Approach . 72

7.1 BatchInstances Attributes for Resource BM's Iterative Pattern 82
7.2 Implementation Statistics for High-Level Batch Nodes 89

Appendix A

Frequent Working Patterns Analysis Results

Monday Shift 1			
Sequence	**Traces with a pattern**	**Traces on a weekday and shift**	**Percentage**
Carga L+D iniciada, Cargado en carro L+D (Together)	11	56	19.64%
Montaje (Together)	10	56	17.86%
Producción montada, Montaje (Together)	15	56	26.79%
Monday Shift 2			
Carga L+D iniciada (Separate)	5	17	29.41%
Carga L+D liberada (Separate)	5	17	29.41%
Carga de esterilizador liberada (Separate)	4	17	23.53%
Carga de esterilizador liberada (Separate) → Carga L+D iniciada (Separate)	3	17	17.65%
Cargado en carro L+D (Separate)	3	17	17.65%
Cargado en carro L+D (Separate) → Carga L+D iniciada (Separate)	3	17	17.65%
Composición de cargas (Separate)	3	17	17.65%
Composición de cargas (Separate) → Carga L+D liberada (Separate)	3	17	17.65%

Montaje, Producción montada (Together)	9	17	52.94%
Monday Shift 3			
Montaje, Producción montada (Together)	14	51	27.45%
Monday Shift 4			
Carga L+D liberada (Separate)	11	50	22.00%
Composición de cargas (Separate)	8	50	16.00%
Entrada Material Sucio (Separate)	16	50	32.00%
Tuesday Shift 1			
Carga L+D iniciada, Cargado en carro L+D (Together)	9	58	15.52%
Carga L+D liberada (Separate)	14	58	24.14%
Carga de esterilizador liberada (Separate)	20	58	34.48%
Montaje (Together)	21	58	36.21%
Producción montada, Montaje (Together)	20	58	34.48%
Tuesday Shift 2			
Carga L+D liberada (Separate)	5	16	31.25%
Cargado en carro L+D (Separate)	3	16	18.75%
Comisionado (Separate)	3	16	18.75%
Montaje, Producción montada (Together)	3	16	18.75%
Producción montada, Montaje, Carga L+D liberada (Together)	3	16	18.75%
Tuesday Shift 3			
Comisionado (Separate)	9	51	17.65%
Montaje, Producción montada (Together)	10	51	19.61%
Tuesday Shift 4			
Carga L+D liberada (Separate)	12	48	25.00%
Composición de cargas (Separate)	8	48	16.67%
Entrada Material Sucio (Separate)	16	48	33.33%
Montaje (Together)	10	48	20.83%
Producción montada, Montaje (Together)	13	48	27.08%
Wednesday Shift 1			
Carga L+D iniciada (Separate)	13	63	20.63%
Carga L+D liberada (Separate)	11	63	17.46%

Carga de esterilizador liberada (Separate)	16	63	25.40%
Composición de cargas (Separate)	12	63	19.05%
Montaje (Together)	19	63	30.16%
Producción montada (Together)	10	63	15.87%
Producción montada, Montaje (Together)	18	63	28.57%
Wednesday Shift 2			
Carga L+D iniciada (Separate)	4	17	23.53%
Carga L+D liberada (Separate)	7	17	41.18%
Carga L+D liberada (Separate) → Comisionado (Separate)	3	17	17.65%
Carga de esterilizador liberada (Separate)	3	17	17.65%
Cargado en carro L+D (Separate)	3	17	17.65%
Cargado en carro L+D (Separate) → Carga L+D iniciada (Separate)	3	17	17.65%
Comisionado (Separate)	4	17	23.53%
Composición de cargas (Separate)	4	17	23.53%
Montaje, Producción montada (Together)	5	17	29.41%
Wednesday Shift 3			
Montaje, Producción montada (Together)	9	54	16.67%
Wednesday Shift 4			
Carga L+D iniciada (Separate)	9	48	18.75%
Carga L+D liberada (Separate)	15	48	31.25%
Carga L+D liberada (Separate) → Entrada Material Sucio (Separate)	9	48	18.75%
Composición de cargas (Separate)	8	48	16.67%
Entrada Material Sucio (Separate)	20	48	41.67%
Producción montada, Montaje (Together)	11	48	22.92%
Thursday Shift 1			
Carga L+D iniciada, Cargado en carro L+D (Together)	10	55	18.18%
Carga L+D liberada (Separate)	16	55	29.09%
Carga de esterilizador liberada (Separate)	22	55	40.00%

Carga de esterilizador liberada (Separate) → Carga L+D liberada (Separate)	9	55	16.36%
Comisionado (Separate)	9	55	16.36%
Montaje (Together)	9	55	16.36%
Producción montada, Montaje (Together)	16	55	29.09%
Thursday Shift 2			
Carga L+D iniciada (Separate)	3	20	15.00%
Carga L+D liberada (Separate)	6	20	30.00%
Carga de esterilizador liberada (Separate)	3	20	15.00%
Comisionado (Separate)	3	20	15.00%
Composición de cargas (Separate)	6	20	30.00%
Composición de cargas (Separate) → Carga L+D liberada (Separate)	4	20	20.00%
Montaje, Producción montada (Together)	7	20	35.00%
Producción montada, Montaje, Carga L+D liberada (Together)	4	20	20.00%
Thursday Shift 3			
Montaje, Producción montada (Together)	8	48	16.67%
Thursday Shift 4			
Carga L+D iniciada (Separate)	7	45	15.56%
Carga L+D liberada (Separate)	9	45	20.00%
Composición de cargas (Separate)	7	45	15.56%
Entrada Material Sucio (Separate)	16	45	35.56%
Producción montada, Montaje (Together)	9	45	20.00%
Friday Shift 1			
Carga L+D iniciada (Separate)	9	50	18.00%
Carga L+D iniciada, Cargado en carro L+D (Together)	8	50	16.00%
Carga L+D liberada (Separate)	16	50	32.00%
Carga de esterilizador liberada (Separate)	21	50	42.00%
Comisionado (Separate)	9	50	18.00%
Composición de cargas (Separate)	11	50	22.00%
Montaje (Together)	12	50	24.00%

Producción montada, Montaje (Together)	19	50	38.00%
Friday Shift 2			
Carga L+D iniciada (Separate)	5	18	27.78%
Carga L+D liberada (Separate)	6	18	33.33%
Carga de esterilizador liberada (Separate)	3	18	16.67%
Cargado en carro L+D (Separate)	3	18	16.67%
Cargado en carro L+D (Separate) → Carga L+D iniciada (Separate)	3	18	16.67%
Composición de cargas (Separate)	6	18	33.33%
Composición de cargas (Separate) → Carga L+D liberada (Separate)	3	18	16.67%
Montaje, Producción montada (Together)	6	18	33.33%
Friday Shift 3			
Montaje, Producción montada (Together)	10	46	21.74%
Friday Shift 4			
Carga L+D liberada (Separate)	13	46	28.26%
Composición de cargas (Separate)	10	46	21.74%
Entrada Material Sucio (Separate)	15	46	32.61%
Saturday Shift 1			
Carga L+D iniciada, Cargado en carro L+D (Separate)	4	14	28.57%
Composición de cargas (Separate)	3	14	21.43%
Saturday Shift 4			
Carga L+D iniciada, Cargado en carro L+D (Separate)	7	13	53.85%
Sunday Shift 1			
Carga L+D iniciada, Cargado en carro L+D (Separate)	4	13	30.77%
Carga de esterilizador liberada (Separate)	5	13	38.46%
Composición de cargas (Separate)	6	13	46.15%
Composición de cargas (Separate) → Carga de esterilizador liberada (Separate)	4	13	30.77%
Sunday Shift 3			

Carga L+D iniciada, Cargado en carro L+D (Separate)	1	1	100.00%
Sunday Shift 4			
Carga L+D iniciada, Cargado en carro L+D (Separate)	6	12	50.00%
Carga de esterilizador liberada (Separate)	2	12	16.67%
Carga de esterilizador liberada (Separate) → Carga L+D iniciada, Cargado en carro L+D (Separate)	2	12	16.67%
Producción montada, Montaje, Carga L+D liberada (Separate)	5	12	41.67%