POLITECNICO DI TORINO

POLITECNICO DI TORINO

Master Degree course in Engineering and Management

Master Degree Thesis

Adaptive Validation of BPMN Exercises: Integrating LLMs for Generating and Assessing Equivalent Solutions.

Supervisors
Prof. Riccardo Coppola
Prof. Giacomo Garaccione

Candidate
Anna Maddalena Alerino

Academic Year 2024-2025

Acknowledgements

Desidero esprimere la mia più sincera gratitudine al mio relatore, **Professore Riccardo Coppola**, per la disponibilità e l'attenzione con cui mi ha seguita durante tutto il percorso di tesi. La sua cortesia, la rapidità con cui ha sempre risposto a ogni dubbio e la chiarezza con cui ha saputo indirizzarmi sono state per me di grande aiuto. Lo ringrazio anche per la sua competenza e per la capacità di rendere ogni confronto un'occasione di crescita, contribuendo in modo decisivo alla qualità e alla solidità di questo lavoro.

Un ringraziamento altrettanto sentito va al mio correlatore, **Professore Giacomo Garracione**, per la costante disponibilità, i preziosi consigli e la guida attenta che mi ha accompagnata nel corso della ricerca. La sua professionalità, unita a una grande gentilezza e capacità di ascolto, mi ha permesso di affrontare con maggiore sicurezza le diverse fasi del progetto e di trovare la giusta direzione nei momenti di incertezza.

Infine, voglio ringraziare di cuore tutte le persone che mi sono state vicine in questi anni. Non è sempre stato un percorso facile, ma il sostegno della mia famiglia, degli amici e dei compagni di università ha reso tutto più leggero. Grazie per avermi incoraggiata, per aver creduto in me e per aver condiviso con me fatiche, risate e soddisfazioni.
Come scriveva Oscar Wilde, *La felicità non è avere quello che si desidera, ma desiderare quello che si ha.* Guardando a questo traguardo, non posso che sentirmi grata per tutto ciò che ho avuto lungo il cammino e per le persone che ne hanno fatto parte.

Abstract

This thesis tackles the problem of offering dependable and scalable support for automated assessment in education by examining the use of Large Language Models (LLMs) in the validation of BPMN (Business Process Model and Notation) exercises. Designing and testing a methodological framework that can produce BPMN models that are structurally compliant and appropriate for methodical evaluation across several quality characteristics is the goal. To provide BPMN-compliant ground realities, a dataset of 24 heterogeneous exercises was created. With the help of an extended taxonomy and specific modelling principles, the approach developed into a two-step pipeline through iterative improvements: organised identification of BPMN elements followed by XML rendering. Overall strong performance was highlighted by evaluation across eight quality parameters, with six exercises attaining the maximum score of 100 and an average score of 93.15. While dimensions like ranking (mean 99.0) and traceability (mean 99.7) showed almost perfect stability, correctness (mean 92.8), completeness (mean 92.8), and consistency (mean 90.4) showed more variability, often falling below 70 for more complex exercises. While certain dimensions, such as ranking and verifiability, operated more independently, correlation analysis demonstrated that these three dimensions are highly interrelated. These results show that BPMN validation frameworks can successfully incorporate LLMs. The suggested strategy highlights the advantages and structural drawbacks of existing models in managing challenging process modelling tasks, while providing a reproducible means of assisting with instruction and evaluation.

Contents

Chapter 1

Introduction

Business Process Model and Notation (BPMN) has established itself as a standard framework for representing organizational processes in a precise yet intuitive way. Its dual nature, combining graphical clarity with formal rigor, makes it a powerful tool for both practitioners and learners. In educational contexts, BPMN is widely used to train students in process thinking and to assess their ability to translate textual descriptions into consistent process models. Ensuring the correctness and quality of such models, however, requires robust validation procedures that go beyond visual inspection. In recent years, automated approaches to BPMN validation have been introduced, offering scalable solutions to support both teaching and assessment.

At the same time, the rapid progress of Large Language Models (LLMs) has opened up new opportunities in domains where natural language and structured logic converge. LLMs are increasingly capable of understanding complex textual input, extracting structured elements, and applying domain-specific rules. Yet their application to BPMN validation remains largely unexplored. The present work positions itself in this space: not as a replacement of existing methods, but as an extension that investigates whether LLMs can reliably support the automatic validation of BPMN exercises. By leveraging their flexibility in interpreting natural language, the thesis aims to test whether LLMs can produce structurally compliant BPMN models and whether these outputs can be systematically evaluated across multiple quality dimensions.

The core objective of the thesis is therefore twofold. First, to design a methodological framework that integrates LLMs into BPMN validation, ensuring that the models produced are both auditable and comparable against authoritative ground truths. Second, to evaluate this framework empirically, analyzing not only the overall performance of the models but also their behavior across specific dimensions of quality such as completeness, consistency, and correctness. The contribution of this work lies in demonstrating how LLMs can be operationalized within a structured evaluation pipeline, highlighting both their strengths and their limitations in addressing structurally demanding tasks.

To achieve this, a heterogeneous dataset of 24 BPMN exercises was collected from publicly

available sources and remodeled into BPMN-compliant ground truths. This dataset was deliberately designed to maximize the variety of BPMN constructs, including branching structures, exception handling, and hierarchical decomposition. On this basis, a two-step pipeline was implemented. The first stage focused on the structured extraction of BPMN elements through iterative prompt refinement, gradually evolving from early experimental designs to a consolidated taxonomy with explicit modeling rules. The second stage translated these structured outputs into valid BPMN XML files, ensuring that results could be directly imported and inspected in standard modeling tools.

The evaluation framework assessed the quality of the generated models across eight dimensions: Correctness, Completeness, Consistency, Traceability, Ranking, Modifiability, Unambiguousness, and Verificability. This multi-dimensional approach allowed for a nuanced understanding of performance. Rather than focusing solely on aggregate scores, the analysis investigated the stability of each dimension, the correlations between them, and the variability across exercises. In doing so, the study was able to identify not only areas of strong reliability, such as traceability and ranking, but also those where the models showed greater fragility, such as completeness and consistency.

The remainder of the thesis is organized as follows. Chapter 2 presents the background, introducing BPMN, its main constructs, and the role of validation in both practical and educational settings. Chapter 3 details the methodology, including dataset construction, the evolution of prompt design, the two-stage pipeline, and the evaluation framework. Chapter 4 reports the empirical results, structured around four research questions addressing overall performance, dimension-level outcomes, stability, and correlations. Chapter 5 concludes the thesis by summarizing the key findings, reflecting on their implications, and outlining possible directions for future research.

Chapter 2

Background

Business Process Model and Notation (BPMN) offers a shared, standard language for describing how work is organized and executed across organizations. Its symbols and rules make process models readable for non-technical stakeholders while remaining precise enough to support analysis, improvement, and, where appropriate, automation. In practice, BPMN is used both in industry and in teaching to cultivate process thinking and to assess the translation of textual requirements into coherent models.

Within this landscape, a variety of validation approaches and tools already help check BPMN models for syntactic compliance and alignment with intended behavior, and they also provide structured feedback to learners. This thesis builds on that foundation and explores a complementary direction: how Large Language Models (LLMs) can be integrated into BPMN-focused workflows in ways that remain faithful to the notation while enabling new forms of structured extraction and evaluation.

This chapter provides the conceptual ground for the study. It first defines BPMN and its core elements; then surveys representative application domains to illustrate why modeling choices matter in practice. It next discusses recurring challenges in BPMN modeling and summarizes how current validation systems address them and where they remain constrained. Finally, it introduces LLMs and the role of prompt engineering as they relate to BPMN tasks, laying the basis for the methodological pipeline developed in the following chapters.

2.1 Definition of BPMN

BPMN (Business Process Model and Notation) is a standardized graphical notation initially developed by the Business Process Management Initiative (BPMI) and now maintained by the Object Management Group (OMG), a non-profit consortium that brings together experts and organizations in the fields of information technology and organizational analysis. Established in 1989, the OMG aims to promote open and interoperable standards, and BPMN, first introduced in 2004, stands as one of its most significant contributions. The notation was designed to provide an intuitive yet semantically rich

visual language capable of representing complex business processes in a structured and comprehensible manner.
Furthermore, unlike other modeling approaches, BPMN is specifically tailored for process-oriented modeling. This focus makes it particularly well suited for applications in process automation, optimization, and business analysis. For example, BPMN diagrams can be used to identify inefficiencies, redundancies, and bottlenecks in existing workflows, enabling organizations to streamline their operations and improve overall performance.
The widespread adoption of BPMN can be attributed to its flexibility and adaptability. The notation supports a wide range of modeling scenarios, from high-level conceptual models to detailed technical specifications. This scalability allows organizations to use BPMN at various stages of the process lifecycle, from initial design and analysis to implementation and monitoring. Furthermore, BPMN's compatibility with Business Process Management Systems (BPMS) enables the direct execution of process models, reducing the gap between design and implementation. This capability has made BPMN an indispensable tool in the field of process automation, where the ability to quickly and accurately translate business requirements into executable workflows is critical.
In addition to its technical advantages, BPMN also plays a vital role in improving communication and collaboration among stakeholders. By providing a common language for describing business processes, BPMN helps align the perspectives of different stakeholders, including business analysts, process owners, and IT professionals. This alignment is particularly important in large organizations, where the complexity of business processes often leads to miscommunication and inefficiencies. The use of BPMN ensures that all stakeholders have a shared understanding of the processes being modeled, thereby reducing the risk of errors and misunderstandings.

2.2 Key elements of BPMN

BPMN provides a rich set of graphical elements that enable the precise and intuitive representation of business processes. This discussion focuses solely on the core and most used elements of BPMN, although the notation includes a significantly larger set of symbols and constructs. These elements can be organized into four main categories: Flow Objects, Connecting Objects, Swimlanes, Artifacts and data.

2.2.1 Flow Objects

Flow objects are the primary building blocks of BPMN diagrams. They represent the key actions, decisions, and events that occur within a business process. Flow objects are further divided into three subtypes: Events, Tasks and Gateways.

Events are critical elements in BPMN that represent occurrences that influence the flow of a process. They can trigger, interrupt, or conclude a process, depending on their type and position within the diagram. Events are categorized into three main types:

- Start Events: These events mark the initiation of a process. They are represented by a single thin circle and can be triggered by various conditions, such as a message

arrival, a timer, or a signal. For instance, in an order fulfillment process, a start event might be triggered by the receipt of a customer order. Examples of start events are start event, timer start event, message start event (Fig. 2.1).

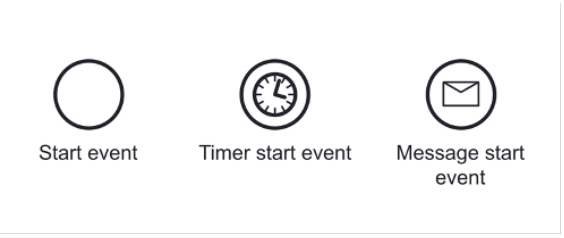

Figure 2.1. Types of start event

- Intermediate Events: These events occur during the execution of a process and can affect its flow. They are depicted as a double thin circle and can represent delays or external triggers. For instance, an intermediate timer event might pause a process until a specific deadline is reached. Examples of intermediate events are message intermediate catch event, message intermediate throw event and timer intermediate catch event (Fig. 2.2).

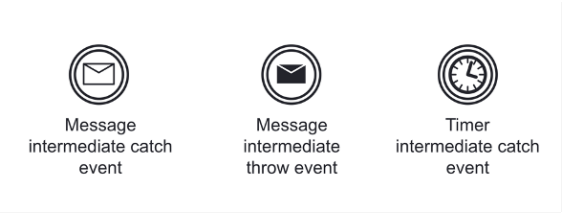

Figure 2.2. Types of intermediate events

- End Events: These events signify the conclusion of a process or a specific process path. They are represented by a single thick circle and can indicate outcomes such as success, failure, or termination. In a customer support process, an end event might represent the resolution of a customer issue. (Fig.2.3)

Tasks are atomic units of work in a process, representing specific actions like "Validate Order" or "Approve Invoice." They can be performed by humans (User Task), automated by a system (Service Task), executed manually without system involvement (Manual Task), run through predefined scripts (Script Task), or used to apply business rules (Business Rule Task)(Fig.2.4). Each type defines how the task is executed and who or what is responsible for it.

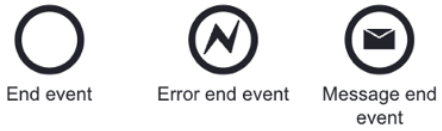

Figure 2.3. Types of end events

Figure 2.4. Types of tasks

A subprocess is a compound activity that encapsulates a set of related tasks and events within a larger process. It can be represented either in a collapsed form (as a single activity with a '+' marker) or in an expanded form (showing its internal flow). Subprocesses are used to simplify complex diagrams, group logically related activities, and improve readability. They may contain their own start and end events, gateways, and intermediate events, and can also host boundary events.

Gateways are decision points that control the branching and merging of process flows. They determine how the process progresses based on conditions, rules, or parallel execution. Gateways are represented as diamond-shaped symbols and are classified into 3 main types: exclusive gatevays, parallel gateways and event-based gateways (Fig.2.5).

- Exclusive Gateways (XOR): These gateways evaluate conditions to select a single path from multiple alternatives. Only one outgoing flow is activated based on the evaluation. For example, in an order approval process, an exclusive gateway might route the order to either a "Fast-Track Approval" or a "Standard Approval" path, depending on the order value.
- Parallel Gateways (AND): These gateways split the process flow into multiple parallel paths, all of which are executed simultaneously. They are also used to synchronize parallel flows before merging them back into a single path. For instance, in a product development process, parallel gateways might be used to initiate concurrent tasks such as "Design Prototype" and "Conduct Market Research".
- Event-Based Gateways: These gateways direct the process flow based on external events, conditions or rules. The process waits for one of several possible events to

occur, and the first event that happens determines the path taken. For example, in a customer support process, an event-based gateway might route the flow based on whether a customer responds to an email or calls the support center.

Figure 2.5. Types of gateway

2.2.2 Pools and Lanes

Pools and lanes are a fundamental organizational feature of BPMN that provide a clear and structured way to distinguish responsibilities and roles within a business process. They help visually separate and categorize the activities performed by different participants, ensuring that the process model is both comprehensible and actionable. Pools represent distinct participants in a business process, such as organizations or departments, and are depicted as large rectangles spanning the diagram. They are essential for modeling interactions between independent entities. Within each Pool, the production process is schematized by including tasks, gateways, and other workflow elements. Lanes, or Swimlanes, are subdivisions within Pools that categorize responsibilities by roles or departments, ensuring a clear assignment of tasks. They are indispensable for modeling complex processes that involve multiple participants or require detailed role-based task allocation. By clearly delineating responsibilities, Swimlanes enable stakeholders to understand who is accountable for each activity and how different entities or departments interact. For example, in a supply chain process, a "Supplier" pool might include Lanes such as "Order Processing" and "Shipping" (Fig.2.6).

Figure 2.6. Example of pools and lanes

2.2.3 Artifacts and data

Artifacts are supplementary elements within a BPMN diagram that contribute to the overall understanding and documentation of a process, though they do not directly influence the execution of the process itself. These elements serve to improve clarity, ensure transparency, and provide additional context for process components. Data elements in a BPMN diagram represent the information that is created, processed, stored, or exchanged throughout a business process. These elements ensure that relevant data flows are accurately depicted, contributing to the coherence and completeness of the model. While BPMN defines multiple types of artifacts, this section focuses specifically on annotations as an artifact and data objects and data stores as the primary data-related elements. These are the most commonly used in practice, providing essential support in modeling real-world processes by clarifying information flows and offering additional descriptive context.

Annotations, the most used artifact in BPMN diagrams, provide descriptive information to clarify the purpose, functionality, or constraints of process elements. Although they do not affect process execution, they are essential for ensuring that all stakeholders, from process designers to business analysts, fully understand the model. By improving documentation and communication, annotations bridge the gap between technical representation and user-friendly process descriptions. Data objects represent the information required or produced during a business process, making explicit how data flows between tasks. They clarify what inputs are needed to complete a process step and what outputs result, facilitating a structured approach to data handling. Used in conjunction with tasks, they illustrate how information is created, processed, and transferred within the workflow. Data stores serve as repositories for persistent data throughout a process, allowing long-term storage and retrieval. They are critical in workflows that require data access across different stages of the process, such as maintaining customer records, financial transactions, or inventory levels. Their presence in BPMN diagrams highlights how information is stored, updated, and retrieved, ensuring process continuity and compliance with data management standards. Fig.2.7 provides a visual representation of the main BPMN artifacts highlighting their respective symbols.

Figure 2.7. Types of artifacts and data

2.2.4 Connecting Objects

Connecting Objects are essential components of BPMN that establish relationships and define the flow of interactions between various elements in a process model. They ensure the logical coherence and continuity of the process by linking Flow Objects, Swimlanes, and Artifacts. Connecting Objects are categorized into three primary types: Sequence Flows, Message Flows, and Associations.

- Sequence flow are the fundamental connectors in BPMN, representing the logical progression of activities within a process. Depicted as solid arrows with a filled arrowhead, they ensure that tasks, events, and gateways follow a structured and predictable sequence. Unlike Message Flows, which indicate communication between different participants, Sequence Flows operate within a single Pool and dictate the order of execution.
- Message flow represents the exchange of information between separate entities, such as different organizations or departments. They are depicted as dashed arrows with an open arrowhead and must always connect elements in different pools. Unlike Sequence Flows, which dictate execution order, Message Flows model communication without enforcing task dependencies.
- Associations provide additional context by linking Artifacts, such as data objects or annotations, to tasks or events. Unlike Sequence or Message Flows, Associations do not influence execution but serve as a means of clarification. They are represented by dotted lines with an open arrowhead, highlighting their supplementary nature.

2.3 Applications of BPMN

In today's fast-paced and interconnected world, the ability to effectively manage and optimize processes is no longer a luxury but a necessity for organizations striving to maintain competitiveness and operational excellence. Across industries, the challenges of complexity, compliance, and efficiency require innovative solutions that can adapt to evolving demands while ensuring transparency and accountability. From the stringent requirements of regulated sectors to the dynamic nature of e-Commerce and from the transformative potential of education to the critical needs of healthcare, the applications of process management tools are vast and varied.
This chapter explores four distinct yet interconnected domains where structured process management plays a pivotal role. Each application highlights unique challenges and opportunities, demonstrating how a standardized approach can drive efficiency, foster innovation, and address systemic inefficiencies. In the following sections, we will delve into the impact on regulated sectors, the role in e-commerce system integration, the advancements in education, and the untapped potential in healthcare.

2.3.1 Impact on regulated sectors

In regulated sectors, the business process model and notation play a crucial role in ensuring compliance, transparency, and accountability. For instance, BPMN plays a crucial

role in the financial sector, enabling organizations to streamline their operations and stay competitive in a rapidly evolving market [1]. Financial services companies face increasing pressure to implement new processes quickly to meet various requirements, including speed-to-market, service quality, and compliance. This need for agility is further complicated by the growing complexity of the financial landscape. By adopting an integrated approach to business processes, financial institutions can ensure that their products, processes, systems, and underlying applications can evolve rapidly to meet changing demands. This integration is particularly vital in key financial operations such as loan provision, insurance policy setup, and investment instruction execution. Optimizing these sale-to-fulfillment processes not only attracts new business but also strengthens customer loyalty and reduces operational costs. Conversely, a lack of integration across various financial services can create vulnerabilities, providing more efficient competitors with significant profit opportunities. Thus, implementing effective Business Process Management solutions is essential for financial institutions to maintain their competitive edge and operational efficiency in today's dynamic market. Moreover by implementing BPMN-based models within a BPMS, organizations can translate process diagrams into automated workflows, reducing the need for manual intervention in repetitive tasks. This integration is particularly beneficial in areas such as loan approvals, customer onboarding, and invoice processing, where automation ensures greater speed, accuracy, and consistency. With BPMS handling execution and monitoring, businesses can optimize resource allocation, enforce compliance, and continuously refine their operations based on real-time performance data.

2.3.2 Impact on E-Commerce

Business Process Model and Notation (BPMN) has become an essential framework for modeling, orchestrating, and optimizing complex workflows within e-commerce ecosystems. Modern digital commerce architectures involve the interaction of heterogeneous microservices, such as order management, inventory control, payment handling, and logistics coordination, and BPMN provides a visual and semantic structure for representing these processes in a coherent and standardized way.
By adopting BPMN, organizations can formalize workflows that improve operational efficiency, reduce errors, and improve adaptability to market fluctuations. In particular, its ability to define events, conditions, and gateways enables the modeling of complex scenarios such as refunds, timeouts, stock recalculations, or exception management. Furthermore, BPMN supports real-time monitoring and fault-tolerant execution, which are crucial for order fulfillment, inventory synchronization, and service-level verification.
Several studies have underlined the benefits of BPMN in the context of e-Commerce and related domains. Zhao and Zou [2] propose a model-driven approach for generating e-commerce user interfaces directly from BPMN process specifications, demonstrating that process information can be leveraged to automatically derive both functional and usability requirements. Likewise, Respício and Domingos [3] extend BPMN with quantitative reliability metrics and apply the Stochastic Workflow Reduction method to evaluate how the failure of individual activities propagates through a process. Their framework enables the computation of the overall reliability of a business process by aggregating the

probabilities of task success or failure. Although developed in a general context, their approach is particularly relevant for e-commerce, where critical workflows, such as payment authorization, delivery tracking, or refund handling, must guarantee transactional integrity and high availability across distributed systems.

In e-commerce, therefore, BPMN offers not only a clear and shared representation of process logic but also a foundation for quantitative assessment (e.g., reliability, performance) and simulation-based scenario testing. Leveraging BPMN for the orchestration of digital operations thus represents not merely a standardization effort but a strategic enabler of operational resilience, controlled scalability, and continuous analytical capability.

2.3.3 Impact on the healthcare sector

Healthcare institutions are inherently complex environments, marked by interdependent workflows, high stakes decision making, and the need for seamless coordination among multidisciplinary teams. Inefficiencies in such settings stemming from poorly defined processes, fragmented procedural guidelines, and inconsistent access to standardized protocols often lead to operational delays, human errors, and compromised patient outcomes. These challenges underscore the critical need for robust frameworks to model, analyze, and optimize institutional processes. BPMN offers a pathway to address these issues by fostering clarity, standardization, and adaptability in workflows and it emerges as a pivotal tool, providing a visual, standardized language to map processes, enhance cross-departmental communication, and mitigate ambiguities in execution.

Despite its theoretical promise, BPMN adoption in healthcare remains limited, as noted in prior research [4], [5]. Barriers such as cultural resistance to process formalization, insufficient training among clinical staff, and reliance on unwieldy textual manuals hinder its integration. This gap reflects a broader tension between the growing demand for operational transparency, driven by digital transformation and quality assurance imperatives, and the slow uptake of innovative process management tools. Standardizing healthcare processes through BPMN could not only align disparate departments but also enable data driven risk management and continuous improvement, transforming procedural inefficiencies into opportunities for systemic refinement. Against this backdrop, Luciano's study [6] offers a timely empirical exploration of BPMN's applicability in healthcare. By surveying 131 professionals across Portuguese health institutions, the research highlights a striking paradox: while only 20.6% of respondents were familiar with BPMN, 72.2% endorsed its potential to streamline workflows and reduce errors. This dissonance underscores both the urgency of adopting structured process management tools and the need to address knowledge gaps among frontline staff.

2.3.4 Impact on education

The study "Gamifying Business Process Modeling Education: A Longitudinal Study" [7] provides a comprehensive exploration of the role of gamification in enhancing the teaching and learning of Business Process Model and Notation within an academic context. The

research investigates the application of the BIPMIN platform, a gamified tool designed to support BPMN education, in a university-level Information Systems course. By integrating game-like elements such as levels, leaderboards, instant feedback, and avatar customization, the study evaluates how these features can foster student engagement, improve motivation, and enhance learning outcomes in the context of BPMN modeling. The findings reveal that gamification plays a significant role in facilitating the comprehension of BPMN modeling principles, leading to a measurable reduction in common modeling errors and an improvement in the accuracy and quality of student-generated diagrams. The structured and immediate feedback provided by BIPMIN enables students to iteratively refine their models, which fosters a deeper understanding of both the syntax and semantics of the notation. This iterative process not only helps students identify and correct mistakes but also encourages a more active and reflective approach to learning.

Furthermore, the study highlights that the gamified approach increases student persistence, as the interactive and engaging nature of the platform motivates learners to continue practicing and improving their skills. The competitive elements, such as leaderboards, and the personalized feedback mechanisms, such as avatar reactions, contribute to a more immersive and enjoyable learning experience. The research also underscores the importance of integrating gamification across the entire course curriculum, as opposed to limiting it to specific exercises, to ensure a cohesive and consistent learning experience. Overall, the study demonstrates that gamification can be a powerful tool in business process modeling education, offering significant benefits in terms of student engagement, motivation, and performance. These findings suggest that the adoption of gamified tools like BIPMIN can transform traditional educational approaches, making complex topics like BPMN more accessible and enjoyable for students while simultaneously improving their ability to apply these concepts in practical scenarios.

2.4 Challenges in BPMN

Despite its widespread adoption and recognized effectiveness in business process management, BPMN presents several challenges that organizations must navigate to fully leverage its potential. Among the most common issues are those related to complexity, consistency, usability and semantic ambiguity. However, these challenges can be managed effectively if properly addressed.

One of the most significant difficulties stems from the inherent complexity of BPMN notation. BPMN offers a vast array of elements, symbols, and modeling capabilities designed to capture business processes with precision. While this comprehensiveness enhances its descriptive power, it can also become a significant impediment, particularly when applied to large-scale or highly intricate workflows [8]. The sheer number of symbols, connectors, and rules can overwhelm stakeholders, especially those without formal training. With over one hundred distinct symbols representing various process elements such as events, activities, gateways, and flows, BPMN facilitates highly detailed process representations. However, this level of granularity often results in models that are difficult to interpret,

particularly for non-technical stakeholders such as business analysts or end-users. Excessive intricacy can lead to misunderstandings or misinterpretations, increasing the risk of inefficiencies in process management.

The practical application of BPMN is fraught with challenges, particularly in maintaining model quality and consistency, as evidenced by the study conducted by Leopold, Mendling, and Günther [9]. One of the most prominent issues identified is the structural inconsistency in process models, particularly in the use of splits and joins. The study reveals that many modelers struggle with distinguishing between parallel and exclusive splits, often leading to deadlocks and multimerges, which can significantly impair the model's functionality and clarity. This problem is exacerbated by the fact that BPMN offers multiple representational choices for the same semantics, a concept known as 'concept excess', which can confuse practitioners and lead to errors in modeling. Additionally, the study highlights the frequent misuse of message flows, particularly the confusion surrounding the throwing message event, which is often misunderstood due to its active nature in contrast to the passive perception of most events. This misalignment between the modeler's intent and the actual representation can result in incorrect process flows, further complicating the model's accuracy.

Another critical challenge lies in the decomposition of models. The study found that many models were excessively large, often exceeding the recommended size for readability, which indicates a lack of proper decomposition. This issue is compounded by inconsistent role descriptions between main processes and subprocesses, leading to semantic mismatches that can disrupt the overall process architecture. This issue reflects a broader problem in process modeling: the tendency to focus on individual models rather than considering their place within a larger, company-wide process framework. This fragmented approach can lead to inconsistencies and redundancies, undermining the integrity of the process architecture.

The usability of BPMN models in practical applications is often limited by the complexity of the notation. Although BPMN is designed to be executable within Business Process Management Systems (BPMS), overly intricate diagrams can hinder their translation into functional workflows. In practice, this complexity may lead to delays or errors during implementation, as developers struggle to interpret and operationalize the models. A common example can be found in healthcare processes, where numerous exceptions and conditional paths create diagrams that are syntactically correct but too cumbersome to automate. In such situations, organizations are frequently required to simplify or refactor their models before deployment, increasing the overall effort needed to implement BPMN effectively.

Beyond these structural and implementation challenges, the complexity of process modeling introduces additional difficulties in interpretation. Research by Reijers and Mendling [8] in the IEEE Transactions on Systems, Man, and Cybernetics highlights the multiple factors that influence process model understandability, with structural complexity

playing a central role. Their study reveals that as models grow in size and connectivity, the cognitive load imposed on stakeholders increases significantly, making comprehension more challenging. This complexity is particularly evident in models with a high number of interconnections, dense control flow structures, and heterogeneous connectors, which can obscure the overall process logic.

Furthermore, different stakeholders,including technical experts and business managers, may interpret the same visual representation in substantially different ways, especially when dealing with intricate control structures such as nested splits and joins. The absence of a clear decomposition mechanism in many modeling notations exacerbates this issue, making it difficult for non-experts to identify related process components. These challenges underscore the need for careful design choices to enhance model readability and facilitate effective communication across diverse user groups.

Despite these considerable challenges, the complexity of BPMN notation is not an insurmountable obstacle. With appropriate training, standardized modeling guidelines, and a concerted focus on simplicity and clarity, organizations can mitigate these issues and fully harness the benefits of BPMN. A tiered approach to modeling, where high-level models are utilized for strategic planning and detailed models are reserved for specific use cases, can help balance complexity and readability. Additionally, fostering a culture of collaboration between technical and non-technical stakeholders, coupled with investments in training and best practices, can enhance comprehension and promote consistency in BPMN application across the organization. By addressing these challenges proactively, businesses can ensure that BPMN remains a powerful and effective tool for process management, enabling them to streamline operations, improve efficiency, and facilitate better decision-making.

2.5 Limitations of Current BPMN Validation Systems

Validation systems for BPMN are indispensable in ensuring the syntactic correctness, semantic coherence, and operational reliability of process models. These systems operate through multi-layered verification mechanisms, addressing both structural conformity to BPMN standards and alignment with domain-specific business logic. Syntactic validation ensures adherence to the formal rules of BPMN, such as the proper use of elements, connectors, and flow structures, as defined by the Object Management Group. Semantic validation, by contrast, focuses on logical consistency, verifying that process flows accurately represent intended business scenarios and avoid ambiguities or contradictions.

Advanced validation frameworks often integrate domain-specific constraints, enabling organizations to enforce compliance with regulatory requirements or internal policies. For instance, tools like BIPMIN incorporate rule-based engines to evaluate diagrams against predefined reference solutions, identifying discrepancies in business entities, actors, and process sequences [7]. Such systems not only detect errors but also provide granular feedback, such as syntax error lists and visual diagram annotations, to guide users in

refining their models. The research demonstrated that students found the tool engaging and beneficial in learning BPMN fundamentals, particularly due to its ability to provide immediate and structured feedback.

2.5.1 Solution Creation in BPMN Validation Systems

One of the main limitations of current BPMN validation systems is the heavy reliance on manual solution definition. Traditional frameworks require instructors or experts to specify all acceptable variants of a process model in advance. This approach is time-consuming and restricts scalability, as every possible equivalent representation must be anticipated. As a result, the validation process often becomes inefficient, especially when applied to large datasets or dynamic environments. Moreover, the necessity of creating multiple representations of the same process model exacerbates cognitive load and resource allocation, leading to inefficiencies in educational and industrial applications.

Even though the BIPMIN platform was demonstrated to be extremely beneficial to students, still several limitations in the evaluation mechanism of BIPMIN were identified. The platform still requires instructors to define a set of reference solutions against which student-generated models are assessed. This constraint limits flexibility, as alternative yet functionally equivalent process representations may not always be recognized as correct. Additionally, students reported that minor variations in terminology or sequencing sometimes led to lower accuracy scores, even when their models were logically sound. This indicates that while BIPMIN introduces a more dynamic and interactive approach to BPMN validation, it does not fully resolve the need for predefined solutions.

2.5.2 Challenges in Textual Analysis for BPMN Validation and Process Generation

The automatic transformation of textual requirements into BPMN diagrams presents several challenges, particularly in the areas of textual analysis, semantic interpretation, and validation accuracy. Despite the increasing use of Natural Language Processing (NLP) techniques, the existing methodologies often struggle to handle linguistic variations, maintain semantic coherence, and ensure structural integrity.

One of the primary challenges in textual analysis for BPMN validation is the dependence on predefined lexical resources. Many existing BPMN transformation systems require manually specified domain-specific terminologies and structural mappings, which significantly limits adaptability [10]. As noted in previous studies, NLP models applied to BPMN generation often fail to account for the diversity of natural language expressions. For instance, variations in wording, synonyms, or different grammatical structures can lead to inconsistencies in process representation. This limitation requires users to anticipate multiple terminological variations, which is impractical for large-scale applications.

Additionally, challenges in semantic interpretation further hinder the reliability of BPMN validation. Automated systems frequently struggle to infer implicit relationships between process elements when such connections are not explicitly defined in the textual input. For example, a BPMN validation system may not recognize that "approve transaction" and "validate purchase" refer to equivalent process steps unless such mappings are manually encoded. This issue underscores the limitations of current NLP techniques in capturing context-dependent variations, leading to reduced accuracy in BPMN diagram generation.

Several approaches have been proposed to address these issues, with a focus on integrating hybrid AI methodologies. A promising strategy involves combining rule-based validation techniques with machine learning models. By leveraging structured representations, such as dependency trees or abstract syntax trees, researchers have improved the accuracy of task identification and sequence determination. Furthermore, advancements in large language models (LLMs), such as fine-tuned GPT-based systems, could be pivotal in extracting structured information from unstructured text, thereby improving the consistency of BPMN diagrams.

2.6 Introducing LLMs for BPMN validation

Large Language Models (LLMs) are advanced artificial intelligence systems designed to process and generate human-like text. Built on deep learning architectures and particularly on the transformer paradigm, they are capable of capturing relationships between words, phrases, and sentences, thereby understanding context and producing coherent responses. Their training on vast and heterogeneous corpora enables them to recognize complex syntactic and semantic patterns, making them valuable tools for tasks that bridge natural language and structured representations. While they have already shown strong results in applications such as text summarization, translation, and reasoning, their potential also extends to domains like BPMN, where structured modeling must often start from textual descriptions.

In the context of BPMN, LLMs are promising because they can interpret unstructured or semi-structured textual inputs and identify the elements that form the building blocks of process models. Descriptions of processes in natural language are rarely uniform: they may vary in terminology, contain implicit steps, or describe activities in ways that are only partially standardized. LLMs can help in bridging this gap by extracting candidate tasks, events, and gateways, and by organizing them into structured formats that facilitate subsequent transformation into formal BPMN diagrams. In doing so, they contribute to reducing the effort required to move from narrative descriptions to standardized representations.

A particularly relevant feature of LLMs in this regard is their ability to handle variability and ambiguity. Natural language is inherently flexible, and the same process logic can often be described in multiple ways. Traditional rule-based approaches tend to struggle

with such variability, frequently rejecting alternative but valid formulations. LLMs, by contrast, can tolerate differences in wording and sequencing, while still preserving the underlying process logic. At the same time, they can be guided to explicitly flag situations where information cannot be deduced from the input text, avoiding silent assumptions and ensuring transparency. This property is especially important in process modeling, where incorrect inferences can propagate into structural inconsistencies.

Another area where LLMs show promise is in supporting standardization and improving model coherence. By learning from large amounts of structured and unstructured data, they are able to internalize modeling conventions, such as the proper use of pools and lanes or the need to match split and join gateways of the same type. While they cannot replace formal validation mechanisms, they can serve as a complementary tool that reduces common errors and suggests more consistent ways of representing logic. In this sense, they can contribute to interoperability across models developed by different individuals or organizations, ensuring that equivalent logic is represented in comparable ways.

Nevertheless, it is important to recognize the limitations that remain. LLMs may generate outputs that are plausible but not correct, reflecting biases present in their training data or making assumptions not supported by the input. They also require significant computational resources and remain, to a large extent, 'black-box' systems whose reasoning paths are difficult to interpret. These challenges highlight the necessity of designing careful protocols for their use, incorporating safeguards to ensure reliability and accountability. Transparency mechanisms, such as explicitly stating when a modeling choice is not deducible from the text, represent an important step in this direction.

In sum, while LLMs are not without risks, their capacity to parse natural language, handle variability, and align outputs with formal standards makes them an intriguing complement to existing BPMN validation approaches. They do not replace traditional frameworks, but they open new possibilities for reducing ambiguity, supporting standardization, and making the transition from unstructured descriptions to formal models more efficient and scalable.

2.7 Prompt engineering for LLMs

2.7.1 Defining Specific Prompts to Guide Models in Understanding Complex BPMN Scenarios

Prompt engineering is a methodological approach designed to optimize the interaction between LLM and specific tasks by carefully crafting input queries. This process ensures that the model generates precise, relevant and structured outputs, significantly enhancing its effectiveness in complex applications. LLMs operate on probabilistic text prediction, meaning that their responses are inherently influenced by the phrasing, specificity, and contextual structure of the input prompts. The primary objective of prompt engineering is to refine the formulation of these inputs to improve the comprehension, accuracy, and

coherence of the response of the model.

In practical applications, prompt engineering serves as an alternative to fine-tuning, allowing users to adapt pre-trained LLMs without requiring extensive additional training. By structuring the instructions to be clear, concise and goal-oriented, it is possible to take advantage of LLMs efficiently across various domains. Best practices in prompt engineering include specifying desired output formats, structuring multi-step queries to break down complex tasks, and iteratively refining prompts to align model behavior with intended outcomes. This iterative refinement is particularly crucial when applying LLMs to structured tasks such as BPMN validation, where precise interpretation of complex workflows is necessary.

Prompt engineering is a fundamental technique in optimizing the effectiveness LLMs for BPMN validation. Due to the complexity and variability inherent in BPMN scenarios, carefully crafted prompts are essential in guiding LLMs to generate precise and contextually relevant responses. The formulation of these prompts involves structuring input queries in a way that maximizes clarity, ensuring that the model correctly interprets the underlying business logic and nuances of BPMN elements.

To achieve this, prompts must be designed with specificity, incorporating well-defined instructions that reduce ambiguity and enhance comprehension. For instance, when prompting an LLM to validate a BPMN process, explicit references to key aspects such as decision gateways, event sequencing, and resource allocation can significantly improve the accuracy of the model's response. Additionally, structured prompts may include examples of correct and incorrect process models, providing a reference framework that enhances the model's ability to distinguish between valid and flawed representations.

Moreover, prompt engineering involves the careful selection of terminologies and phrasings that align with BPMN standards. Since LLMs are trained on diverse corpora, ensuring that prompts employ consistent technical vocabulary mitigates the risk of misinterpretation. Furthermore, iterative refinement of prompt structures allows researchers and practitioners to progressively improve the model's comprehension, ensuring its ability to handle increasingly intricate BPMN scenarios effectively.

2.7.2 Enhancing BPMN Validation via Systematic Prompt Engineering in Large Language Models

The reliability of LLMs in validating BPMN diagrams hinges on the implementation of systematic testing strategies to calibrate model behavior. This research investigates methodologies to optimize prompt design, ensuring consistent and high-quality validation outcomes. Central to this effort is the exploration of prompt engineering techniques that improve interpretative precision and coherence, enabling LLMs to effectively analyze BPMN models composed of interconnected elements (events, tasks, gateways, and sequence flows) which collectively define business process logic.

A critical focus lies in refining prompt language to eliminate ambiguity: structured directive prompts, such as 'Identify all errors in the following BPMN process and explain their impact', yield more comprehensive results than open-ended queries like 'What do you think about this BPMN model?' Explicit instructions reduce vagueness, guiding the model toward logically consistent evaluations. This approach is further augmented by embedding contextual examples within prompts. Annotated BPMN diagrams paired with detailed explanations of errors serve as training signals, enhancing the model's ability to generalize patterns and recognize deviations from standard notation.

Prompt chaining emerges as another pivotal strategy. Prompt chaining is a technique where multiple prompts are linked sequentially to guide a Large Language Model through a structured reasoning process. Each prompt builds on the previous response, refining the model's output step by step. This approach improves accuracy, coherence, and reliability, making it particularly useful for complex tasks like BPMN validation and multistep problemsolving. So, by decomposing validation into sequential stages, first identifying errors and then soliciting corrective recommendations, LLMs generate structured, actionable insights. This stepwise method mirrors human analytical processes, enabling nuanced assessments of workflow logic and compliance with BPMN rules.

To establish best practices, systematic comparative testing evaluates model performance across diverse prompt formulations. Metrics such as accuracy, completeness, and consistency in error detection are analyzed under varied conditions. For example, scenario-based prompts simulate real-world cases, exposing the model to heterogeneous process structures while integrating domain-specific constraints (e.g., regulatory requirements or organizational policies). These scenarios minimize ambiguity, aligning outputs with expected validation standards.

Iterative refinement underpins the entire framework. By continuously testing and adjusting prompts based on performance feedback the aim is to calibrate the model's responsiveness to BPMN syntax and semantics. This process includes enriching prompts with definitions of BPMN elements, relational dynamics between components, and validation rules. Such contextual grounding sharpens the model's ability to distinguish correct workflows from logically inconsistent configurations.

The aim is to use the transformative role of systematic prompt engineering in bridging the gap between LLM capabilities and structured BPMN analysis. The findings advocate for a methodical, evidence-based approach to prompt design, ensuring that LLMs evolve into dependable partners in business process optimization.

2.8 Foundations of BPMN Model Evaluation

The evaluation of Business Process Model and Notation (BPMN) diagrams is a key step to guarantee that process models are not only visually appealing but also logically correct and reliable. In both organizations and educational contexts, BPMN is often used to describe and analyze workflows. However, if models are not systematically assessed, they may contain hidden mistakes, such as inconsistencies or structural errors, that reduce their clarity and practical value.

In the academic literature, different perspectives on model quality have been discussed. Some researchers underline the importance of how well a model reflects the meaning of the real domain or how useful it is for stakeholders. Others focus on the structural layer, arguing that the formal properties of a model, such as correctness and absence of errors like deadlocks, are the necessary foundation before any semantic or practical interpretation can take place [11].

Several approaches have been proposed to support this structural view. Experiments have shown that elements like model size, complexity, and even the way activities are labeled can influence how easily people understand a model and how likely it is to contain errors [12]. In addition, concepts from software engineering have been adapted to process modeling: metrics such as coupling (how strongly different parts of the model are connected), cohesion (how consistent tasks are within the same part of a model), and complexity (how simple or intricate the logic is) have all been suggested as indicators of quality [13].

These metrics are often combined with practical guidelines, such as the Seven Process Modeling Guidelines (7PMG) [11], which provide concrete rules on when a model becomes too large or too complex. Together, they show that structural quality can be measured and, when necessary, improved through targeted revisions. This makes structural validation not only a theoretical concept but also a practical tool to increase the reliability and clarity of BPMN models.

2.9 Foundations of BPMN Model Quality

The quality of a business process model is central to BPMN research and practice. To define this quality, semiotic theory distinguishes three levels of analysis: syntax, semantics, and pragmatics. For BPMN models, this means that a model must first achieve syntactic correctness, using symbols and constructs according to the rules of the notation, before its semantic content can be interpreted and its pragmatic usefulness evaluated [12].

This perspective emphasizes the importance of structural quality. If a model suffers from syntactic or structural flaws, such as incorrect use of gateways or missing synchronization, it becomes difficult to interpret its meaning and unreliable for analysis or execution.

Empirical studies confirm that deficiencies in structural correctness reduce comprehensibility and increase the likelihood of misinterpretation, even when the semantic content is clear [12]. As a result, structural validation has become a critical focus in BPMN research and practice.

Ensuring structural correctness means verifying that the control flow is logically coherent, free of contradictions, and consistent in its use of modeling constructs. This verification serves as a prerequisite for higher-level analysis, such as examining the adequacy of domain terminology or the suitability of the model for communication with stakeholders. Within this context, the concept of soundness plays a particularly important role. A sound BPMN model can always be initiated correctly, progresses through activities without encountering irresolvable conflicts, and reaches a proper completion state. Violations of soundness manifest as structural errors like deadlocks (where the flow is blocked indefinitely) or livelocks (where the process loops without advancing). Other issues, such as inconsistent branching logic or missing start/end events, also contribute to the breakdown of soundness and significantly undermine the model's clarity and reliability [11].

A comprehensive understanding of BPMN model quality requires both qualitative judgment and measurable criteria. Structural metrics play a vital role in this evaluation, allowing for a systematic assessment of the model's size, complexity, and structuredness. Many of these metrics are adapted from software engineering principles and have proven valuable in assessing the correctness and comprehensibility of process models.

2.9.1 Structural Metrics for BPMN Model Evaluation

A fundamental aspect of BPMN model evaluation involves structural metrics, which provide measurable indicators of model correctness and comprehensibility. These metrics are derived from principles in software engineering and focus on aspects such as model size, complexity, and the degree of structuredness in the process design [13].

Model Size is one of the most straightforward metrics, typically measured by the number of nodes or activities in a BPMN diagram. Larger models tend to be harder to understand and maintain because they present more elements to process simultaneously. Empirical research shows that when a model exceeds a certain number of nodes, its error probability and cognitive load significantly increase [11]. To mitigate this, guidelines such as the Seven Process Modeling Guidelines (7PMG) suggest decomposing models that surpass specific size thresholds.

Complexity captures the intricacy of control-flow relations within the model. It is often quantified using the Control-Flow Complexity (CFC) metric, which counts the number of branching and merging structures in the process. Higher complexity correlates with a greater cognitive burden on the model reader and increases the likelihood of design errors [12]. As in software engineering, simpler structures are easier to validate, communicate, and adapt over time, reducing the risk of misinterpretation.

Structuredness refers to the degree to which a model follows well-defined patterns of process decomposition. Structured models use paired splits and joins consistently, making their logic more predictable and their behavior easier to verify. In contrast, unstructured models often include ambiguous paths or incomplete synchronizations, which can lead to behavioral anomalies. Research shows that structuredness enhances model understandability and reduces the likelihood of errors [11].

Together, these metrics provide a solid foundation for evaluating BPMN models. Specific thresholds can guide the evaluation process: models with more than thirty nodes are considered candidates for decomposition, gateways with more than three incoming or outgoing flows are discouraged, and unstructured constructs should be avoided whenever possible. These thresholds transform abstract notions of quality into concrete, operational rules that can be consistently applied across different BPMN models.

2.9.2 The Evaluation Framework for BPMN Models

Table 2.1 illustrates the main structural metrics adopted for the evaluation of BPMN (Business Process Model and Notation) diagrams. These metrics constitute established criteria for assessing model quality, with a particular focus on complexity, comprehensibility, and structural soundness. The inclusion of recommended thresholds and guidelines provides a reference framework for maintaining models at a manageable level of detail, while the accompanying rationale clarifies the implications of each metric in terms of error probability, interpretability, and overall model reliability.

Table 2.1. Structural Metrics for BPMN Evaluation

Metric	**Definition**	**Recommended Threshold / Guideline**	**Rationale**
Model Size (Nodes)	Total number of elements (activities, events, gateways) in the model	Avoid models with >30 nodes; decompose if necessary (7PMG)	Larger models are harder to understand and maintain; higher error probability
Gateway Degree	Number of incoming/outgoing flows connected to a gateway	No more than 3 incoming/outgoing flows per gateway (7PMG)	High degree increases complexity and risk of misinterpretation
Structuredness	Extent to which the model follows structured patterns with paired splits and joins	Model as structured as possible; avoid unstructured constructs	Structured models are more predictable, easier to validate and comprehend
Concurrency	Level of parallelism and simultaneous execution paths in the process	Minimize excessive concurrency; prefer manageable parallelism	Excessive concurrency leads to complexity, synchronization issues, and errors

2.9.3 Automatic Validation Approaches

In addition to manual inspection and the use of structural metrics, research has also advanced towards automatic validation techniques. These methods aim to support modelers by detecting structural issues and suggesting improvements without requiring extensive human intervention.

One widely used approach is the automatic checking of guidelines and thresholds. Established rules, such as limiting the number of nodes in a model or restricting the degree of gateways, can be encoded in validation tools. In this way, models can be scanned automatically, and potential violations are flagged for correction. Empirical studies confirm that such threshold-based checks help to identify models that are more prone to errors or harder to comprehend [11].

Beyond simple checking, researchers have also investigated structural refactoring as a way to improve model quality. The principle is to transform a model into a more structured version without altering its behavior. For instance, replacing unstructured constructs with structured patterns can increase clarity and predictability. However, this process is not without challenges: in some cases, achieving a fully structured model requires duplicating activities, which may itself create confusion or reduce readability [11].

Despite these advances, automatic approaches still face important limitations. Natural language ambiguities make it difficult to automatically refactor textual labels into consistent verb-object structures across languages [11]. Moreover, there is an inherent trade-off between enforcing strict structural correctness and preserving the original modeling intent. As a result, fully automated validation cannot replace human judgment; rather, it complements it by offering systematic checks and suggesting candidate improvements.

In summary, automatic validation techniques contribute significantly to the reliability of BPMN evaluation. By combining guideline checking with refactoring strategies, they extend the reach of quality assurance, while also highlighting the areas where human expertise remains indispensable.

Chapter 3

Methodology

The methodological framework adopted in this study is articulated in eight main activities that build upon one another, starting from the preparation of the dataset and leading to the final analytical evaluations. The process begins with the selection of BPMN exercises and the definition of ground-truth reference models, continues with the use of LLMs for element extraction and XML generation, and proceeds with the definition and application of evaluation metrics. The subsequent steps include the computation of overall and dimension-level scores, followed by the analysis of their stability and inter-dimension correlations.

To provide a clear overview of this process, Figure 3.1 presents a graphical representation of the methodological pipeline. The diagram organizes the eight activities in a structured flow and explicitly highlights their connections to the research questions (RQ1-RQ4). This mapping clarifies how each methodological step contributes to addressing the study's objectives and ensures a transparent link between the design of the methodology and the analyses presented in the results.

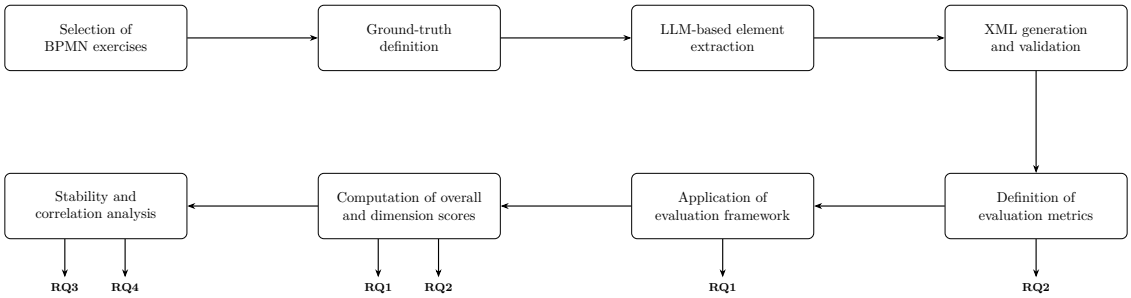

Figure 3.1. Methodological pipeline with vertical mapping to RQ codes.

3.1 Exercise Selection and Construct Definition

The complete list of exercises is provided in Appendix A. The construction of the exercise corpus was guided by a maximum-variation, construct-oriented sampling strategy. Rather than privileging textual length or narrative richness, the primary objective was to assemble a set of tasks that maximized the heterogeneity of BPMN constructs encountered by the modeling pipeline. To this end, 24 exercises were selected, spanning both IT and business domains, with an emphasis on scenarios resembling realistic organizational processes. The exercises were collected from multiple public, English-language websites, and no translation or linguistic adaptation was required. Many items were originally presented as bullet lists, which facilitated stepwise interpretation, while others were in prose and were retained when they contributed unique structural features. Inclusion centered on the presence of salient BPMN constructs, and exercises that were lengthy but structurally trivial were excluded to avoid diluting the construct signal. Given the study's focus on testing the robustness of prompt-based BPMN generation across structurally diverse cases, construct variety rather than text length was the governing criterion. Priority was assigned to exercises exhibiting one or more of the following characteristics:

- control-flow branching and recomposition, such as explicit splits followed by multiple task sequences;
- exception handling and error management, with the presence of exceptional paths or explicit references to error conditions;
- nesting and hierarchical structure, including multi-step procedures that entail subactivities or layered logic.

Selection proceeded iteratively, and for each exercise, both a close reading of the textual description and an inspection of the provided reference solution image were conducted to anticipate the implied BPMN structure. Perceived difficulty was then judged qualitatively on the basis of structural features, such as the depth of nesting, the number and type of gateways, and the presence of exceptions or error paths. No formal coverage quotas, for instance, a minimum count of boundary events or subprocesses, were imposed; instead, the target was balanced variety achieved through purposive judgment. Although the sources furnished solution images for the selected exercises, these depictions did not always conform rigorously to BPMN standard notation. To establish a consistent basis for evaluation, a curated ground truth was produced by remodeling each retained exercise in BPMN using the bpmn.io environment. During this step, targeted notational and semantic adjustments were applied where necessary to enforce BPMN compliance, and the resulting models constitute the authoritative reference (gold) for subsequent assessments. While no numeric quotas were predefined, the selection was intentionally diversified across canonical BPMN families in order to stress-test specific modeling capabilities that are known to be challenging in text-to-BPMN generation. In particular, coverage included:

- **Events** (start, intermediate, boundary, end), with emphasis on error and exception cases, to verify whether the pipeline could correctly infer event-driven behavior from textual cues such as triggers, interruptions, or terminations;

- **Tasks and activity patterns**, including multi-step sequences suggestive of hierarchical decomposition, to assess whether multi-stage procedures were preserved as coherent units rather than flattened or reordered;
- **Gateways**, with particular attention to branching structures that produce multiple post-split task sequences, to evaluate the reconstruction of decision logic and the correct allocation of downstream activities to alternative or parallel paths;
- **Exception and error flows**, whether interrupting, compensatory, or alternative paths, to test the distinction between ordinary alternatives and genuine exception handling, which is a frequent source of ambiguity in prose descriptions.

By covering such a broad range of cases, the methodology repeatedly encountered structural patterns most likely to cause modeling errors, thereby creating a clear testbed to assess both the strengths and the weaknesses of the approach. Pre-existing bullet-point formatting was considered advantageous but not required.
The first seven exercises were initially tested with ChatGPT 4.0, and to ensure consistency and comparability, the entire set was subsequently re-run with ChatGPT 5.0. This alignment of all outputs to the same model increased output solidity, with greater stability in gateway selection and element coverage, thereby enabling a more homogeneous analysis and a more reliable control chain.

3.2 Initial Attempts and Iterative Refinement

This subsection reconstructs the iteration path that led from early, minimal instructions to the consolidated extraction prompt presented in Subchapter 3. We report the design intentions, the failure modes observed at each step, and the methodological decisions that followed. For clarity, we use the version labels Test 1 → Test 2 → Test 3 → Test 4 → Final prompt.
At the very beginning, we attempted to generate a BPMN XML directly from the textual input. When this approach proved unreliable, we set aside the idea of immediate XML conversion. Instead, we decided to create an ordered list of elements because it would be easier to evaluate at the end of the process. Only after finalizing this list-based prompt did we revisit the XML step, finding that the structured list made the final XML more coherent. In other words, the list was a step we introduced along the way to ensure the final output was more easily assessable.

Test 1: End-to-end text (baseline, discarded)

Intent
The initial attempt was to ask the LLM to generate a complete `.bpmn` file directly from a natural language description of the process. The rationale behind this approach was to evaluate whether the model could autonomously produce a ready-to-use diagram, thereby eliminating the need for intermediate steps of representation or transformation.

Emergent limitations:

- **Non-guaranteed importability.** The generated files were frequently not readable by BPMN tools. In many cases, the `.bpmn` output could not be imported at all, or it contained syntax errors that rendered the diagram invalid. This severely limited the usability of the approach, since a process model that cannot be parsed or visualized fails to serve its core purpose.
- **Missing or incorrect sequence flows.** Connectors between tasks and events were often misplaced or inconsistent. Arrows were sometimes drawn in arbitrary directions, paths were interrupted, and logical dependencies were distorted. These issues undermined the reliability of the generated process and created diagrams that were misleading or incomplete.
- **Absent or confused pools.** The model consistently struggled with the representation of pools and lanes. In several cases, they were entirely omitted; in others, the model confused their semantics, failing to distinguish between inter-organizational boundaries (pools) and internal roles or responsibilities (lanes). This compromised one of the fundamental aspects of BPMN: the ability to capture organizational context and responsibilities.

Resulting decision
Given these recurring issues, we concluded that direct XML generation was not a viable baseline. The lack of structural reliability and frequent syntactic errors prevented meaningful evaluation. As a result, we temporarily abandoned this approach and shifted our focus toward developing a prompt that would first extract an *ordered list of process elements*. This intermediate representation proved easier to evaluate, validate, and later transform into a correct `.bpmn` file.

Test 2: Per-element records (without grouped taxonomy)

Intent
After abandoning direct XML generation, the next attempt was to shift toward a more granular representation. Instead of asking the model for a complete diagram, we requested a list of individual records (e.g., Start, End, Gateway, Intermediate, Boundary, Task), each with a minimal set of descriptive fields. This representation improved transparency: for the first time, it became possible to clearly see which elements the model had identified and how it categorized them. Compared to raw XML, the output was therefore easier to read and manually interpret.

Emergent limitations:

- **No grouped taxonomy.** The prompt did not instruct the model to return an ordered and grouped list of BPMN elements classified under fixed headers (Pools, Lanes, Start Events, Tasks, Gateways, Intermediate Events, Boundary Events, End Events). As a result, the output remained flat, and elements of different types could appear mixed or inconsistently sequenced.

- **No gateway pairing rules.** The prompt lacked explicit requirements ensuring that gateway splits and joins followed the same type (e.g., XOR split must rejoin with XOR, AND split must rejoin with AND). This omission frequently produced logically incoherent models.
- **Pools and Lanes misidentification.** The distinction between Pools (independent participants) and Lanes (internal role subdivisions) was not consistently applied, leading to models where organizational boundaries were blurred or incorrectly represented.
- **Insufficient semantic guidance for gateways.** The main gateway types (XOR, AND, OR, Event-based, Complex) were only listed, without clear symbolic notation, behavioral descriptions, or usage examples. This limited the model's ability to apply gateways correctly in different process contexts.

Resulting decision
Given these limitations, the flat per-element listing was judged insufficient for reliable analysis. The absence of grouping and the lack of logical rules for gateways made the format too unstable to evaluate systematically. To address these gaps, the next iteration introduced an *ordered and grouped taxonomy* with fixed headers and consistency requirements for gateways. This shift marked a decisive move toward standardization, ensuring that the output could be checked more systematically and later reused as a solid basis for further steps.

Test 3: Ordered and grouped list with Pools/Lanes rules

Intent
Building on the per-element records of Test 2, this iteration introduced two key innovations aimed at improving both structure and interpretability:

i) An *ordered and grouped taxonomy* of BPMN elements, requiring the output to be organized under fixed headers (Pools, Lanes, Start Events, Tasks, Gateways, Intermediate Events, Boundary Events, End Events).

ii) Explicit *structural constraints* on Pools and Lanes, clarifying that Pools are always mandatory, Lanes optional, and that sequence flows cannot cross Pool boundaries.

Improvements obtained:

- **Standardized grouping.** By adopting a fixed taxonomy, the output could now be checked systematically for both completeness and consistency, overcoming the "flat" and less reliable representation used in Test 2.
- **Clearer yet imperfect Pool/Lane coherence.** The introduction of formalized rules for Pools and Lanes, together with the prohibition of cross-Pool flows, enhanced the representation of organizational boundaries and collaboration. Nevertheless, inconsistencies still occurred, albeit less frequently than before. This issue turned out to be the hardest to address and persisted, in various forms, throughout the entire prompt refinement process.

- **Gateway policy introduced.** This iteration also added interpretive rules: consistent split–join pairing, allowance for intermediate elements between gateways, and a heuristic to interpret lists of sub-activities "within the same step" as parallel actions when no sequencing or conditional indicators were present.

Residual issues:

- **Conditional logic errors.** The model continued to confuse XOR versus OR, and parallel versus sequential logic, particularly in terse or ambiguous textual descriptions.
- **Pool–Lane misuse.** In some cases, the model generated multiple Pools and connected them using normal sequence flows, effectively treating Pools as if they were Lanes and thus breaking collaboration rules.
- **Inconsistent adherence to guidance.** Despite the additional structural rules, the model often failed to apply them consistently:
  - it missed the "parallel list" heuristic (e.g., interpreting lists such as a), b), c) within the same step → Parallel Gateway);
  - it ignored the **intermediate flow allowance**, forcing unnecessary adjacent gateways;
  - it inconsistently applied the **Pools mandatory / Lanes optional** rule when filling element records.

Resulting decision
Given these residual issues, we decided to further tighten interpretive guidance by adding a high-level requirement: the model must explicitly detect conditional ("IF") structures and decision points before assigning gateways. The details of this refinement are presented in the next subsection.

Test 4: Reinforcement on Conditional ("IF") Logic

Intent
Building on the residual weaknesses identified in Test 3, this iteration focused specifically on improving the handling of conditional logic. A prescriptive block entitled *Key Requirement – Analysis of Conditional (IF) Logic* was introduced, requiring the model to:

- explicitly identify decision points in the text and map them to the appropriate gateway type (XOR, OR, AND, Event-based);
- disambiguate compact lists that could conceal conditional structures versus genuine concurrency;
- avoid speculative interpretations of branching that were not grounded in the source text.

Improvements obtained:

- **More consistent decision handling.** The additional requirement reduced ambiguity in gateway selection. The model became more reliable in distinguishing between exclusive and inclusive choices, as well as between parallel and sequential flows.

Residual issues:

- **Persistent Pool–Lane confusion.** The model continued to blur the distinction between Pools (representing separate participants) and Lanes (representing internal responsibilities within a Pool).
- **Lack of subprocess representation.** The taxonomy still did not include a dedicated category for subprocesses, limiting the ability to represent hierarchical process structures.

Resulting decision
These unresolved issues—namely Pool/Lane misuse and the absence of subprocess support—were carried forward as priorities to be addressed in the final consolidated prompt.

Final Prompt - Consolidation (taxonomy + rules + two-stage pipeline)

In this section, only the main changes introduced with respect to the previous version have been summarized. The complete text of the final prompt, together with a detailed commentary on its structure and rules, will be discussed later in the chapter.
After the progressive refinements carried out in Tests 1–4, the methodology reached its final stage with the consolidation of all improvements into a single comprehensive prompt. This final version integrated the lessons learned from earlier iterations, addressing residual shortcomings such as the lack of subprocess support, persistent Pool/Lane ambiguities, and the absence of explicit ambiguity-handling. The resulting prompt represented a stable and standardized framework, suitable both for systematic evaluation and for reuse in subsequent steps of the pipeline.

Key additions:

- **Extended taxonomy.** A new group dedicated to Subprocesses was added, with its own identifiers (SP1, SP2, ...) and required attributes. Each subprocess must specify its name, origin Pool (and Lane if present), reference ID, and previous element. The prompt also clarified whether the subprocess is collapsed or expanded, and allowed the attachment of boundary events. This extension closed a gap identified in Test 4, enabling hierarchical representation within the same framework.
- **Clearer collaboration discipline.** The Pools vs. Lanes distinction—still problematic in Test 4—was explicitly defined. Pools are described as distinct participants (organizations, systems, external actors), across which no sequence flows are allowed; Lanes are optional subdivisions within a Pool, representing roles or departments. Flows can cross Lanes within one Pool but not Pool boundaries. The

prompt also provided explicit guidance for single-lane cases, avoiding the creation of artificial or redundant structures.

- **Ambiguity management.** To avoid silent invention or over-interpretation, an explicit ambiguity-handling protocol was added. Whenever a required attribute or modeling choice could not be inferred from the text, the output had to state *not deducible from the text* in the relevant field. This ensured transparency, marked underspecified aspects of the source text, and enabled evaluators to distinguish genuine model errors from faithful reflections of textual ambiguity.

Methodological rationale. With these additions, the prompt reached its final form. At this stage, the methodology also consolidated the two-stage pipeline: (i) extraction of a structured, ordered list according to the taxonomy, followed by (ii) rendering into XML. This separation between semantic parsing (language understanding, classification, flow logic) and artifact synthesis (XML and layout) was not part of the prompt text itself, but a methodological choice derived from the experimental path. It stabilized the process, reduced inconsistencies, and made verification faster and more reliable.

3.2.1 Comparative Overview of Prompt Evolution

To better visualize the progressive refinements of the prompt, Table 3.1 provides a comparative overview of its evolution across the different testing phases. Each iteration introduced specific additions or methodological adjustments aimed at addressing previously identified shortcomings. The table highlights, for every version, the main objectives and innovations, the issues that were effectively resolved, and the residual limitations that remained. This structured comparison clarifies how the methodology incrementally converged towards the final prompt.

Table 3.1. Comparative Overview of Prompt Evolution

Version	**Main objective / additions**	**Issues resolved**	**Residual issues**
Test 1	Direct XML generation from text (`.bpmn` file)	—	Importability failures; missing/incorrect sequence flows; absent or confused Pools/Lanes.
Test 2	Per-element records with minimal fields (Start, End, Gateway, Task, etc.)	Element traceability; improved transparency of model's interpretation	No grouped taxonomy; no gateway pairing rules; insufficient semantic guidance for gateways.
Test 3	Ordered & grouped taxonomy; explicit Pools/Lanes rules (Pools mandatory, Lanes optional); gateway policy (pairing, intermediate flow allowance, parallel list heuristic)	Standardized, machine-checkable format; clearer organizational boundaries; more consistent gateways	Errors in conditional logic (XOR vs. OR, parallel vs. sequential); misuse of Pools as Lanes; inconsistent adherence to interpretation tips.
Test 4	Addition of "Key Requirement – Analysis of Conditional (IF) Logic"	Better disambiguation of decision points; more consistent gateway allocation	Pools vs. Lanes confusion persisted; no dedicated group for Subprocesses; no explicit ambiguity-handling protocol.
Final Prompt	Extended taxonomy (added Subprocess group); clarified semantic distinction Pools vs. Lanes (participants vs. roles); ambiguity management protocol ("not deducible from the text"); methodological adoption of two-stage pipeline (extraction → XML)	Subprocess support; definitive Pool/Lane clarification; explicit ambiguity handling; structural stability	—

3.3 Final Prompt for Ordered Element Extraction

This subsection reproduces the final prompt in its entirety, consolidating definitions, scope, naming rules, and interpretation criteria into a single instruction set. The output serves as the canonical intermediate artifact for all downstream steps. Immediately afterward, we provide a detailed commentary articulating its logic, taxonomy, constraints, and the rationale behind each design choice.

Final Prompt for Ordered Element Extraction

PROMPT
Return an ordered and grouped list of the BPMN elements present in the process, classified according to the following main types:

- Pools
- Lanes
- Start Events
- Tasks
- Gateways
- Intermediate Events
- Boundary Events
- End Events
- Subprocesses

`THIS IS THE TEXT "`
Key Requirement - Analysis of Conditional ("IF") Logic:
Carefully examine whether the text includes any conditional structures or decision points.
Interpretation tip:

- Parallel Gateway
  When there is a list of multiple sub-activities (e.g., a), b), c)...) within the same step or bullet point, assume that a Parallel Gateway is likely involved to model these actions as concurrent tasks.
- Gateway pairing rule:
  If flows split using a Parallel Gateway (AND) or an Exclusive Gateway (XOR), and then rejoin later in the process, they must be merged using the same gateway type:
  A Parallel Gateway split must be followed by a Parallel Gateway join
  An Exclusive Gateway split must be followed by an Exclusive Gateway join
  Mixing gateway types (e.g., splitting with XOR and merging with AND) creates logical inconsistency.

- Intermediate flow allowance
  Between two gateways (of any type), there can be one or more tasks or other BPMN elements. The gateways do not need to be immediately adjacent.

Pools and Lanes (Structural Rules)

- Pools are mandatory
  Every BPMN element must specify the name of the pool to which it belongs (even if the pool is collapsed or contains a single lane).
  Pools represent participants in a process (e.g., an organization, a company, or an external system). They define clear boundaries between different process owners or entities. Each pool is independent, and communication across pools requires explicit message flows.

- Lanes are optional:
  If lanes are present, the name of the lane must be specified, and each element must indicate which lane it belongs to.
  Lanes exist only within a pool. They are subdivisions used to organize and categorize activities by role, department, or responsibility. Unlike pools, lanes do not create process isolation: flows can freely cross lanes within the same pool, but not between pools.

For each BPMN element, ensure the following fields are present:
Start event:

1. Choose one of the following types:
   - Message Start Event
     Description: The process starts upon receipt of a message from an external entity (e.g. a request from a customer).
   - Timer Start Event
     Description: The process starts at a specified time or time interval (e.g. every Monday at 9:00 a.m.).
   - Conditional Start Event
     Description: Start occurs when a certain condition (expressed by a rule or variable) is true.
   - Signal Start Event
     Description: Starts when a broadcast signal is received, useful for starting several processes in parallel.
2. Assign a name
3. Provide the name of the pool (and the lane if present) of origin (collapsed or not)
4. Provide the reference identifier (start from SE1 for each start event)
5. Previous element (refer to the numbers of the reference identifiers)

6. Pool name (required) and lane name (if present)

End event:

1. Choose one of the following types:

   - None End Event
     Description: Simply indicates the end of the process, with no specific effects or outputs.
   - Message End Event
     Description: At the end of the process, a message is sent to an external actor (e.g., order confirmation to the customer).
   - Error End Event
     Description: Ends the process by generating an error, which can be caught by an error handler (usually in a subprocess).
   - Escalation End Event
     Description: Signals an escalation, useful to inform a higher level or to trigger an alternative process handling.
   - Compensation End Event
     Description: Triggers compensation mechanisms to "repair" the effects of previous activities (e.g., refunding a customer after an error).
   - Signal End Event
     Description: Broadcasts a signal at the end of the process, which can be received by other processes.
   - Terminate End Event
     Description: Immediately terminates the process and all its active instances, including parallel ones.

2. Assign a name
3. Provide the name of the pool (and the lane if present) of origin (collapsed or not)
4. Provide the reference identifier (start from EE1 for each end event)
5. Previous element (refer to the numbers of the reference identifiers)
6. Pool name (required) and lane name (if present)

Task:

1. Task Label Name
2. Types of tasks:

   - User Task
     Activity manually performed by a human user through an interface (e.g., filling out a form).

- Manual Task
  Fully manual activity, without system support (e.g., physically moving an object).
- Service Task
  Activity performed by an automated service or application (e.g., API call).

3. Provide the Name of the Pool (and if there are multiple lanes, also the corresponding lane) of Origin (Collapsed or not)
4. Provide the Reference Identifier (If there are multiple tasks, each must have a number: start from T1, then T2, etc., according to the number tasks)
5. Previous Element. (Refer to the reference identifier numbers)
6. Pool name (required) and lane name (if present)

Gateway (indicate if multiple flows exit):
They can have one or more flows in input or output

1. Choose one of the following types:
   - Exclusive Gateway (XOR)
     Allows only one outgoing path among several alternatives, based on a condition. Only one path is followed.
   - Inclusive Gateway (OR)
     Can activate one or more outgoing paths simultaneously, if the corresponding conditions are true.
   - Parallel Gateway (AND)
     Starts all branches in parallel or waits for all branches to converge. Does not evaluate conditions.
   - Complex Gateway
     Handles custom logic (e.g., at least two out of three paths).
   - Event-Based Gateway
     The flow is triggered based on the event that occurs first.
2. Assign a name
3. Provide the name of the pool (and lane if present) of origin (collapsed or not)
4. Provide the reference identifier (If there are multiple gateway, each must have a number: start from G1, then G2, etc., according to the number gateways)
5. Previous element (refer to the numbers of the reference identifiers)
6. Pool name (required) and lane name (if present)

In BPMN (Business Process Model and Notation), gateways are elements used to control the flow of a process, especially for managing decisions, parallel paths, conditional flows, and synchronization.
Main Types of Gateways:

1. Exclusive Gateway (XOR) – Diamond (empty or with an X)
   Allows only one outgoing path to be taken based on conditions.
   Example: 'If payment is approved, continue. Otherwise, send a notification.'
   Symbol: ♦ or ♦X
   Behavior: only one flow proceeds

2. Parallel Gateway (AND) – Diamond with a '+' symbol
   Activates all outgoing paths simultaneously.
   Also used to synchronize multiple incoming flows.
   Symbol: ♦+
   Behavior: all flows start or must complete

3. Inclusive Gateway (OR) – Diamond with an 'O'
   One or more outgoing flows can be taken, depending on conditions.
   Symbol: ♦O
   Behavior: multiple flows may proceed in parallel, but not necessarily all

4. Event-based Gateway – Diamond with a thin circle
   Waits for an external event (e.g., message received) to determine which path to follow.
   Symbol: ♦O
   Behavior: flow proceeds based on which event occurs first

5. Complex Gateway – Diamond with an asterisk
   Used for advanced flow logic, e.g., 'activate 3 out of 5 paths.'
   Symbol: ♦*

Usage Examples:

- Use an Exclusive Gateway for an if/else type of decision.
- Use a Parallel Gateway to start multiple tasks at the same time.
- Use an Inclusive Gateway when multiple conditions may be true.
- Use an Event-based Gateway when flow should wait for a user action or external event.

Intermediate event:

1. Type of intermediate event:
   - Message intermediate catch event
     Waits for the reception of a message from an external entity before continuing the process.

- Message intermediate throw event
  Sends a message to an external entity as part of the process.
- Timer intermediate catch event
  Pauses the flow until a specific date/time or time interval.
- Escalation intermediate throw event
  Generates an escalation to signal a situation requiring higher-level attention.
- Conditional intermediate catch event
  Waits for a specific condition to be met before proceeding.
- Link intermediate catch event
  Entry point for an internal link within the diagram (useful for avoiding crossed lines).
- Link intermediate throw event
  Exit point for an internal link within the diagram.

2. Assign a name
3. Provide the name of the pool (and if there are multiple lanes, also the corresponding lane) of origin (collapsed or not)
4. Provide the reference identifier (if there are multiple intermediate events, each must have a number, starting from IE1, then IE2, ...)
5. Previous element (refer to the numbers of the reference identifiers)
6. Pool name (required) and lane name (if present)

Boundary Events (attached to tasks):

1. Types of boundary events:
   Boundary events can be interrupting (interrupt the activity) or non-interrupting (do not interrupt it). Both versions are shown in the images.

   - Message boundary event
     Reacts to the reception of a message during the execution of an activity.
   - Timer boundary event
     Reacts to a timeout while an activity is in progress.
   - Escalation boundary event
     Signals an escalation while the activity is being executed.
   - Conditional boundary event
     Reacts to the occurrence of a specific condition.
   - Error boundary event
     Catches an error generated during the activity.
   - Signal boundary event
     Reacts to the reception of a broadcast signal.

- Compensation boundary event
  Triggers a compensation activity if needed.
- Message boundary event (non-interrupting)
  Handles an external message without interrupting the activity.
- Timer boundary event (non-interrupting)
  Starts an alternative time-based flow without interrupting the activity.
- Escalation boundary event (non-interrupting)
  Triggers an escalation without interrupting the activity.
- Conditional boundary event (non-interrupting)
  Reacts to a condition without stopping the ongoing activity.
- Signal boundary event (non-interrupting)
  Responds to a signal without interrupting the associated activity.

2. Assign a name
3. Provide the name of the pool (and if there are multiple lanes, also the corresponding lane) of origin (collapsed or not)
4. Provide the reference identifier (if there are multiple boundary events, each must have a number, starting from BE1, then BE2, ...)
5. Element to which it is attached (refer to the numbers of the reference identifiers)
6. Pool name (required) and lane name (if present)

Subprocess:
A subprocess is a compound activity that encapsulates a set of related tasks and events within a larger process. It can be used to group and simplify complex logic.

1. Types of subprocesses:

   - Embedded Subprocess
     Contained within the parent process, it cannot exist independently. It is used to structure and group activities without external reference.
   - Reusable (Global) Subprocess
     A standalone process that can be invoked from multiple parent processes. It promotes reuse and consistency.
   - Event Subprocess
     Triggered by a start event (message, timer, conditional, etc.) while the parent process is running. Can be interrupting or non-interrupting.
   - Transaction Subprocess
     Used when the activities inside must follow a transaction protocol (commit or rollback).
   - Ad-hoc Subprocess
     Contains activities that may occur in an undefined sequence, optionally requiring completion conditions.

2. Important considerations for subprocesses:
    - Decide to use a subprocess when a group of activities logically belongs together and would otherwise clutter the main diagram.
    - Each subprocess must specify:
        - A clear name
        - The pool (and lane if present) of origin
        - A reference identifier (start from SP1, SP2,)
        - The previous element (refer to the numbers of the reference identifiers)
        - Whether it is collapsed (shown as a single activity box with a [+] marker) or expanded (showing internal flow).
    - Boundary events can be attached to subprocesses, just like to tasks.
    - A subprocess may contain its own start events, end events, gateways, and other elements, following the same structural rules as the parent process.
    - Using a subprocess should improve clarity, not add unnecessary complexity. It should only be chosen if it provides semantic grouping or reusability.

Ambiguity Management and Guardrails:
If any required attribute or modeling choice cannot be directly inferred from the text, explicitly state "not deducible from the text" in the relevant field. This ensures that any ambiguity is clearly marked and that no element is invented or assumed beyond what is provided. Remember that the model must not introduce pools, lanes, activities, or events that are not supported by the text. Each task should have only one outgoing sequence flow, and all elements must respect the fixed taxonomy and identification scheme provided.

3.3.1 Prompt Design and Characteristics

The prompt has been designed with the overarching objective of producing a deterministic, exhaustive, and auditable list of BPMN elements that appear explicitly or can be unambiguously inferred from the exercise text. This objective responds to the methodological necessity of extracting structured information from unstructured descriptions in a manner that is both rigorous and transparent. The construction of the prompt is therefore guided by three fundamental principles:

- Semantic fidelity
- Structural standardization
- Traceability

The first principle, semantic fidelity, ensures that the extracted list does not simply mirror the sequence of sentences in the source text but instead captures the actual control-flow semantics implied by the narrative. This guards against superficial readings and guarantees that the reconstructed model remains faithful to the logical intentions expressed in

the prose. The second principle, structural standardization, requires that every identified element be reported according to a fixed schema, thus guaranteeing that outputs from different exercises are directly comparable and readily consumable by automated tools. Finally, the third principle, traceability, mandates that each entry be associated with a minimal set of relational information, most importantly the "previous element." This ensures that the logical continuity of the process is preserved and that the dataset can subsequently be translated into XML without the need to return to the original exercise text. In combination, these principles ensure that the resulting lists are semantically accurate, structurally uniform, and logically coherent.

The taxonomy underlying the prompt has been deliberately kept narrow in order to focus on the essential building blocks of process modeling while excluding extraneous artifacts. All elements must be assigned to one and only one of the following groups: Pools, Lanes, Start Events, Tasks, Gateways, Intermediate Events, Boundary Events, End Events, and Subprocesses. This categorical framework provides clarity and prevents ambiguity in classification, while at the same time ensuring that all the information necessary to reconstruct a valid process model is included. Ancillary or decorative aspects of BPMN, such as artifacts not directly tied to control flow, are deliberately omitted in order to maintain focus on the central aim of semantic evaluation. The outcome is a dataset in which each identified item is situated within a precise and standardized taxonomy that covers the full spectrum of core control-flow constructs.

In addition to defining the taxonomy, the prompt prescribes a uniform reporting schema that must be followed for every element. Each entry requires a stable reference identifier composed of type-specific prefixes and incremental numbering, a descriptive name or label derived directly from the text, and a mandatory indication of pool (together with lane, where applicable). The schema also mandates the inclusion of the "previous element" field, which serves as an anchor to capture the immediate upstream logic of the flow. This field is maintained consistently across all element types, even for start events where it is explicitly marked as not applicable. The identifier convention is particularly significant because it employs separate counters for different families of elements, such as tasks, gateways, events, and subprocesses, thereby ensuring that identifiers remain both human-readable and programmatically robust. This approach not only avoids collisions but also allows for unambiguous cross-referencing in subsequent stages of XML synthesis, providing a reliable bridge between textual analysis and executable process models.

The interpretation of control flow and gateway selection occupies a central role in the prompt, given that natural language descriptions often compress complex branching logic into succinct formulations. The prompt explicitly requires a careful scan for conditional structures and synchronization points and imposes rules to ensure consistent modeling choices. Two constraints are of particular importance:

- Homogeneous pairing
- Concurrency recognition

The first constraint, homogeneous pairing, dictates that when a split is modeled using a particular family of gateways, the corresponding join must employ the same family, even if the intervening process flow contains multiple layers of activities or other elements. This ensures internal consistency and prevents mismatches in branching logic. The second constraint, concurrency recognition, is designed to capture situations in which the text enumerates multiple sub-activities that are intended to occur independently. In the absence of explicit sequencing or conditional cues, such enumerations are interpreted as parallel tasks. Event-based decisions are handled as a separate case, focusing on the selection among awaited signals or external triggers, and the introduction of unsupported gateway types is strictly prohibited. By anchoring gateway modeling to clear linguistic cues, the prompt minimizes discretionary interpretation and enhances the repeatability of results across different exercises.

Events and subprocesses are also treated according to standardized rules in order to avoid inconsistencies. All event families, start, intermediate, boundary, and end, must be recognized and reported explicitly, with boundary events requiring clear notation of their attachment. When the text specifies whether a boundary event is interrupting or non-interrupting, this distinction must be recorded in the output; when such information is absent, the relevant field is retained but marked according to the ambiguity protocol. Subprocesses, in turn, are understood as semantic devices for grouping and hierarchical encapsulation. For each subprocess, the schema requires a dedicated identifier, an explicit location in terms of pool and lane, and a link to the surrounding flow via the "previous element" field. When the text specifies whether the subprocess is collapsed or expanded, this status must also be recorded. In this way, the prompt preserves hierarchical structures and makes subprocess boundaries explicit wherever the narrative supports them, while avoiding artificial or unjustified decompositions.

A rigorous protocol is applied to the management of ambiguity and uncertainty. Whenever a required attribute or modeling choice cannot be deduced from the text, the output must explicitly state "not deducible from the text." This practice performs a dual function: it alerts the reader to the limitations of the available information, and it ensures that subsequent analyses can distinguish between errors in modeling and faithful reflections of the source's underspecification. In addition, the prompt enforces a series of guardrails that prohibit the introduction of unsupported pools, lanes, activities, or events; maintain the rule that every task may only have a single outgoing sequence flow; and require adherence to the fixed taxonomy and identifier scheme. Together, these measures guarantee that the resulting dataset remains transparent, free from silent invention, and consistent with the principles of reproducible research.

Finally, in terms of its methodological placement, the prompt represents the culmination of an iterative refinement process and has been intentionally designed as a stand-alone extraction stage, distinct from the task of diagram rendering. This separation serves two major purposes. First, it isolates the delicate work of semantic parsing and classification from subsequent tasks such as XML generation and graphical layout, thereby avoiding

confounding factors linked to tool-specific representations or layout algorithms. Second, it produces a stable and auditable intermediate representation of process logic that can be directly inspected by human researchers and systematically compared against curated ground truth models. By functioning as this intermediary layer, the prompt becomes the keystone of the overall methodology: it bridges unstructured natural language text and structured process models, allowing for evaluation that is transparent, replicable, and comparable across a diverse and heterogeneous set of exercises.

3.4 Prompt for BPMN XML Generation

Below is reproduced, verbatim, the final prompt used to synthesize a BPMN 2.0-compliant XML (.bpmn) diagram from the ordered element list produced with the prompt present in Subchapter 3. The rendering stage does not merely adjust graphical layout; it also includes a logical consistency check. If the model contains inconsistencies that cannot be represented in BPMN, the rendering phase intervenes with targeted logical adjustments. These modifications are minimal and localized, ensuring that the core process logic remains intact while making the diagram fully BPMN-compliant. It specifies file structure, layout discipline, identifier policy, and guardrails. Immediately afterward, a detailed commentary that explains its objectives, scope, constraints, and the rationale behind each requirement is given.

PROMPT
Generate a BPMN 2.0 XML file that includes full graphical layout information (BPMN-DI), ensuring the result can be opened, visualized, and interacted with in modeling tools such as Camunda Modeler, Signavio, or Visual Paradigm. The output must be provided as a .bpmn file ready for download.
Technical Requirements The generated file must include:

- All BPMN elements described in the input: start events, intermediate events, end events, user/manual/service tasks, gateways, sequence flows, and others if specified.
- A clear and readable graphical layout, with elements automatically positioned on a regular grid (horizontal or vertical).
- Consistent spacing between all BPMN elements (minimum 150px horizontally, 100px vertically).
- A complete BPMN-DI section, including:
  - `<BPMNShape>` for each BPMN element with correct coordinates and dimensions.
  - `<BPMNEdge>` for each sequence flow, with waypoints that respect visual logic and ensure directional arrows are correctly rendered in tools.

Sequence Flow Rendering Rules (BPMNEdge Waypoints) Waypoints must follow these rules:

- Flows always start from the right edge of the source element.
- Flows always end at the left edge of the target element.
- The edge path must contain at least two waypoints, forming a straight or right-angled line.
- The order of waypoints must follow the direction of the flow, from source to target.

Special rules:

- Gateways (diamond): flows must enter from the left vertex and exit from the right vertex.
- Events (circle): flows must enter from the left side and exit from the right side.
- Tasks (rectangle): flows must enter from the left center and exit from the right center.
- Sequence flows must not overlap or cross any other element (use bends if necessary).

Pools and Lanes Requirements

- All pools and lanes described in the input must be created and explicitly represented in the BPMN XML.
- Each pool must include its defined lanes using the `<laneSet>` structure.
- Each lane must correctly reference the BPMN elements (tasks, events, gateways) that belong to it.
- The BPMN-DI section must include `<BPMNShape>` definitions for pools and lanes, with accurate coordinates and sizes.
- Positioning must guarantee that each lane is visually nested inside its pool, and that all flow elements are placed inside the correct lanes.
- Layout must avoid overlaps or misalignments between pools and lanes, ensuring a clean, structured diagram.
- In case of a single lane inside a pool, the lane name may be placed directly next to the pool instead of nesting.

Input format Below this prompt, the BPMN process will be described using sequential IDs and clear names for all elements. Based on that input, please generate:

- The `<process>` block with all BPMN elements and their sequence flows.
- The `<BPMNDiagram>` block with all `<BPMNShape>` and `<BPMNEdge>` tags.
- A .bpmn file that can be downloaded and directly opened in Camunda Modeler, displaying the full diagram with correctly rendered connections.

The input is the following: “

The XML prompt operationalizes a rendering-only stage whose input is exactly the ordered and grouped list obtained thanks to the prompt present in Subchapter 3 (IDs, names, pool, lane if present, and a single Previous element anchor). Sequence flows are reconstructed strictly from these Previous element anchors, ensuring one-to-one correspondence between the list and the serialized XML.

3.4.1 Prompt Design and Characteristics

The XML prompt was designed with three main objectives in mind:

- Determinism
- Separation of concerns
- Auditability

Determinism applies strictly to control-flow semantics and element identifiers. While the logical structure of the BPMN remains identical for the same input list, minor deterministic adjustments in the diagram's layout may be applied to ensure BPMN-DI compliance. These layout changes do not alter the underlying process logic. Separation of concerns restricts the rendering stage to faithfully transcribing the already validated list into a BPMN-compliant XML structure, without reinterpreting the source text or making semantic decisions. This isolates rendering from semantic parsing, making it clear whether errors originate in the list extraction or in the rendering itself. Auditability guarantees that each exercise produces a single BPMN file containing only file content and no natural-language commentary. This strict discipline supports automated import tests, schema validation, and reproducible re-runs, while ensuring that downstream analyses always compare like-for-like artifacts.

Scope and Input Contract. The prompt assumes the upstream artifact includes the fixed groups (Pools, Lanes, Start Events, Tasks, Gateways, Intermediate Events, Boundary Events, End Events, Subprocesses), family-scoped identifiers (SE, T, G, IE, BE, EE, SP), pool/lane placement, and the Previous element for each node.

Core Serialization Requirements. The prompt specifies that a complete BPMN file must contain:

- Single `<bpmn:definitions>` element
- Processes and laneSets for each pool
- Flow nodes matching the ordered list
- Sequence flows reconstructed from Previous element anchors
- `<bpmndi:BPMNDiagram>` with shapes and edges

This ensures that every pool is represented by its process and, where applicable, a `<bpmn:laneSet>` referencing contained nodes. Flow nodes match the element families in the ordered list, with boundary events attached to their host via `attachedToRef`. Sequence flows are strictly reconstructed from Previous element anchors and gateway policy, ensuring each flow references valid `sourceRef` and `targetRef`. The file also contains a `<bpmndi:BPMNDiagram>` block with shapes (`<bpmndi:BPMNShape>` + `<dc:Bounds>`) for each node and edges (`<bpmndi:BPMNEdge>` + `<di:waypoint>`) for each sequence flow.

Identifier and Naming Policy. The XML prompt enforces a deterministic strategy for identifiers and names:

- Family prefixes
- Name attributes
- Auxiliary artifacts

Family prefixes ensure that each element preserves the same prefix used in the ordered list (e.g., SE for Start Events, T for Tasks, G for Gateways, IE for Intermediate Events, BE for Boundary Events, EE for End Events, SP for Subprocesses). These are combined with incremental numbers to ensure global uniqueness (e.g., SE1, T2, G3). Name attributes must replicate exactly the labels provided in the ordered list, transferred verbatim into the XML without reinterpretation. Auxiliary artifacts, such as sequence flows, lane containers, and DI shapes, are deterministically generated with systematic identifiers derived from the IDs of the nodes they connect (e.g., `sf_T1_T2`, `shape_T1`). This guarantees stability across repeated runs, prevents collisions, and ensures traceable cross-references throughout the file.

Layout and BPMN-DI Discipline. The XML prompt enforces a grid layout with:

- $\geq$150 px horizontal spacing
- $\geq$100 px vertical spacing
- Type-appropriate shapes with stable bounds
- Orthogonal routing of sequence flows

Each node receives a shape with stable coordinates, and each sequence flow receives a BPMN-DI edge with ordered waypoints, ensuring directionality (from right edge of the source to left edge of the target). Pools and lanes are represented with adequate padding, avoiding overlaps and ensuring legibility. This aspect marked a turning point in the methodology, since early attempts at direct text-to-XML generation failed due to unusable layouts. The XML prompt therefore specifies detailed and unambiguous rules for diagram geometry and BPMN-DI encoding, ensuring that rendered diagrams are both valid and readily interpretable by humans and tools.

Output Discipline and Acceptance. The prompt requires file content only, with no natural-language commentary. Acceptance is defined as successful import into standard BPMN editors without manual fixes. Structural coherence is mandatory: every BPMN node has a corresponding DI shape, every sequence flow has a DI edge, and all references resolve correctly.

Observed Behavior During Rendering (Consistency Safeguard) During XML generation, the model occasionally performed lightweight internal consistency checks on

the ordered list (for example, verifying that diverging branches could be rejoined or that boundary attachments were representable) and minimally adjusted rendering to maintain BPMN representability. These adjustments are documented as a pragmatic safeguard, with the ordered list from Subchapter 3 remaining the authoritative source. Substantive discrepancies are flagged, not silently corrected. The rendering prompt is therefore conservative, privileging representability while preserving traceability.

Methodological Rationale. By separating rendering into a dedicated prompt with explicit serialization and layout discipline, the methodology localizes diagram-level issues, avoids reinterpreting textual semantics at this stage, and provides a deterministic, import-ready artifact. This preserves the audit trail from text → ordered list → XML and supports reproducible, tool-agnostic evaluation across the entire exercise set.

3.4.2 Definition of Criteria

Based on the theoretical foundations discussed, a set of evaluation criteria was defined to systematically assess the quality of BPMN models. The criteria were designed to reflect different dimensions of model quality, ranging from syntactic correctness to semantic clarity and structural organization. Each criterion was grounded in the literature, but also adapted to ensure applicability to the dataset of exercises under analysis. This subsection presents the main criteria included in the framework.

Correctness. Correctness refers to the compliance of a process model with the syntactic rules of BPMN. A correct model respects the formal grammar of the notation, ensuring that events, activities, and gateways are properly connected and that start and end events are appropriately defined. Previous studies (e.g., [13]) emphasize that correctness is a prerequisite for any further evaluation, as syntactic errors compromise both executability and interpretability.

Completeness. Completeness assesses whether all relevant aspects of a process are represented in the model. In the context of BPMN, this includes the presence of key events, tasks, and decision points that are necessary to capture the intended process logic. Incomplete models risk misrepresenting the underlying process and may mislead stakeholders. The criterion draws from work on process model comprehension [12], where missing elements were found to reduce interpretability and accuracy in analysis.

Consistency. Consistency concerns the uniformity of modeling choices within a single model. This dimension captures whether gateways are properly paired (e.g., split and join of the same type), whether message flows are used correctly across pools, and whether role assignments remain coherent across subprocesses. Inconsistent models introduce ambiguity, as different parts of the model may contradict each other or apply rules differently. Consistency is also linked to the broader concept of reliability in model-based analysis.

Traceability. Traceability refers to the extent to which each element of a BPMN model can be unambiguously linked back to the original textual description or specification from which it was derived. In educational and validation contexts, traceability is essential to ensure that every activity, event, or gateway has a clear justification in the source material, avoiding the introduction of arbitrary elements. Transparent traceability enhances the auditability of models and reduces the likelihood of hidden assumptions that may distort the intended logic. A model with high traceability allows evaluators to systematically verify that the diagram faithfully reflects the input requirements.

Ranking. Ranking captures whether the relative order of activities in the BPMN model correctly reflects the intended sequencing as described in the source. While correctness focuses on syntactic compliance, ranking emphasizes the preservation of logical and temporal priorities across tasks. Errors in ranking may not necessarily break syntactic validity but can compromise the semantic fidelity of the model, leading to misinterpretation of process execution. Previous studies on model alignment suggest that even minor misorderings can impair understandability, especially in educational contexts where learners are expected to reproduce the correct flow of actions.

Modifiability. Modifiability evaluates how easily a BPMN model can be adapted, extended, or revised without introducing inconsistencies. From a software engineering perspective, modifiability is a key maintainability property, and its adaptation to BPMN highlights the importance of modular design and manageable complexity. Highly entangled structures with excessive cross-dependencies reduce modifiability, as small changes risk propagating unintended side effects. In contrast, models designed with clear modular boundaries and structured subprocesses facilitate iterative refinement and make the model more resilient to evolving requirements.

Unambiguousness. Unambiguousness assesses whether the BPMN model avoids semantic or structural ambiguities that could lead to multiple interpretations. Ambiguities may arise from unclear gateway configurations, inconsistent labeling, or overlapping message flows that obscure the intended logic. Ambiguity significantly increases the cognitive load of readers, as stakeholders may interpret the same structure in divergent ways. This criterion therefore captures the degree to which a model communicates a single, consistent process logic, supporting reliable analysis and execution.

Verificability. Verificability concerns the extent to which the BPMN model can be systematically checked for compliance with predefined rules, constraints, or reference solutions. A verifiable model is one whose structure and semantics lend themselves to automated or semi-automated validation, ensuring that logical soundness and conformity can be tested without requiring ad-hoc interpretation. This dimension is particularly relevant in educational applications, where objective evaluation depends on the ability to reliably compare student-generated models against authoritative ground truths. High verificability thus supports reproducibility, transparency, and fairness in assessment.

Taken together, these eight criteria provide a comprehensive framework for evaluating BPMN model quality. By combining structural aspects (Correctness, Completeness and Consistency) with semantic and communicative dimensions (Unambiguousness, Traceability, Ranking, Modifiability, and Verificability), the framework ensures that assessment is not limited to syntactic compliance but extends to clarity, reliability, and practical usability. This multidimensional perspective allows for a more nuanced understanding of model strengths and weaknesses, highlighting not only whether a model is formally correct but also whether it is intelligible, adaptable, and verifiable in practice. In this way, the criteria collectively establish a robust foundation for both automated evaluation and human-centered analysis, bridging technical rigor with educational relevance.

3.4.3 Excel Implementation

To translate the evaluation criteria into a practical and replicable assessment tool, the framework was operationalized through a structured Excel sheet (The complete Excel file used for evaluation, containing the detailed scoring across all exercises and criteria, is provided as a digital annex to this thesis.). This choice was motivated by the need for clarity, transparency, and efficiency in managing the evaluation of multiple BPMN exercises. Excel offered the flexibility to represent each criterion explicitly, while also ensuring that results could be aggregated and compared across exercises.

The overall structure of the Excel sheet was designed to mirror the logic of the framework. Each row of the Excel sheet corresponds to one evaluation dimension (e.g., Correctness, Completeness, Consistency, Traceability, Unambiguousness). The columns capture the information associated with each dimension, including the dimension label, the numerical score assigned, and a textual explanation of each dimension entry and comments justifying the score. This matrix-like structure enables a systematic organization of both quantitative and qualitative evidence.

The dataset is organized across multiple sheets: 24 dedicated to the individual BPMN exercises and four additional sheets used for aggregated analyses. This structure allows evaluators to examine each exercise in detail while also enabling synthesis at the dataset level. Vertically, the layout makes it possible to observe the treatment of all dimensions within a single exercise, while horizontally, it permits a comparison of how a specific dimension is assessed across exercises. In this way, the Excel file supports both micro-level evaluation and macro-level analysis, ensuring transparency and reproducibility of the assessment process.

The sheet was also designed to support subsequent aggregation and analysis. By assigning numerical values to each criterion, it became possible to compute average scores per exercise, aggregate results per dimension, and identify patterns across the dataset. This functionality was crucial for linking the evaluation framework to the research questions presented at the end of the methodology. In addition, the explicit representation of each criterion facilitated reproducibility: the evaluation procedure can be replicated by other researchers using the same dataset, yielding comparable results.

Finally, the implementation in Excel provided a pragmatic balance between rigor and usability. On the one hand, it maintained a strong connection with the theoretical principles derived from the literature. On the other hand, it offered a simple and accessible tool for applying those principles to real BPMN exercises. This ensured that the evaluation process was not only academically grounded but also practically feasible, paving the way for the systematic analysis presented in the results chapter.

3.5 Research Questions

To connect the methodological framework with the subsequent analysis, the study was structured around a set of explicit research questions (RQ). These questions provide a clear link between the activities described in the methodology and the empirical results, ensuring that the evaluation remains focused and systematically organized.

- **RQ1:** What are the overall scores obtained across exercises?
- **RQ2:** How do the results vary across the evaluation dimensions (e.g., correctness, completeness, consistency, etc)?
- **RQ3:** How stable are the scores across different exercises, and what patterns of variability can be observed?
- **RQ4:** What correlations exist between the evaluation dimensions, and how do they interact across exercises?

These research questions serve a dual purpose. First, they establish the analytical focus of the study by decomposing the evaluation into distinct yet complementary perspectives. Second, they provide a logical bridge between methodology and results: each research question corresponds to a specific set of analyses, enabling the reader to navigate seamlessly from the methodological design to the empirical findings. As such, the RQs function as a guiding structure for both the presentation and the interpretation of results.

Chapter 4

Results

4.1 RQ1: What are the scores obtained by automated analysis of BPMN exercises?

4.1.1 Overall Scores per Exercise

The overall scores for all 24 exercises are reported in Table 4.1. Scores range from 69.25 (Ex13) to 100.00, a value achieved in six different cases (Ex2, Ex5, Ex11, Ex19, Ex22, and Ex24). The mean score is 93.15, the median is 95.56, and the standard deviation is 8.20. As shown in Table 4.1, the majority of exercises performed at a high level: 17 out of 24 obtained scores equal to or above 90, indicating a strong concentration in the upper range. Within this group, a notable subset reached the maximum score, reflecting instances of complete and structurally consistent outputs. By contrast, a limited number of exercises scored lower: four exercises are in the 85-90 range (Ex10, Ex17, Ex21, Ex15), while three fall below 85 (Ex12 = 81.74; Ex18 = 75.83; Ex13 = 69.25). The distribution is therefore characterized by a compact upper cluster and a small left-hand tail driven by a few weaker cases.

Table 4.1. Overall scores per exercise

Exercise	Score	Exercise	Score	Exercise	Score
Ex1	97.05	Ex9	98.32	Ex17	88.74
Ex2	100.00	Ex10	88.82	Ex18	75.83
Ex3	94.87	Ex11	100.00	Ex19	100.00
Ex4	95.73	Ex12	81.74	Ex20	98.76
Ex5	100.00	Ex13	69.25	Ex21	89.64
Ex6	95.40	Ex14	92.81	Ex22	100.00
Ex7	98.28	Ex15	85.00	Ex23	93.74
Ex8	98.80	Ex16	92.78	Ex24	100.00

4.1.2 Distribution of Overall Scores

The overall distribution of scores across the 24 exercises is illustrated in Figures 4.1 and 4.2.
The visualization confirms that the vast majority of cases fall within the upper part of the scale: in particular, more than half of the exercises are concentrated in the 95–100 interval. The middle ranges appear less populated, with only a limited number of exercises falling within the 85–90 band. A very small subset, specifically three exercises, record scores below 85, while at the lower extreme a single case (Ex13) anchors the 65–70 interval, marking the weakest observed performance.
Figure 4.1, by contrast, displays the same results disaggregated by exercise, allowing a direct comparison of individual performances. The chart shows a consistently high level of quality across the majority of exercises, with only a handful of outliers deviating from the overall trend. Taken together, the two visualizations highlight two important aspects: first, the results are strongly skewed toward the upper end of the scale, confirming the overall reliability of the outputs; second, the small left-hand tail indicates that a minority of exercises exhibited more pronounced difficulties, thereby creating a clear separation between the bulk of high-performing cases and a limited number of weaker outliers.

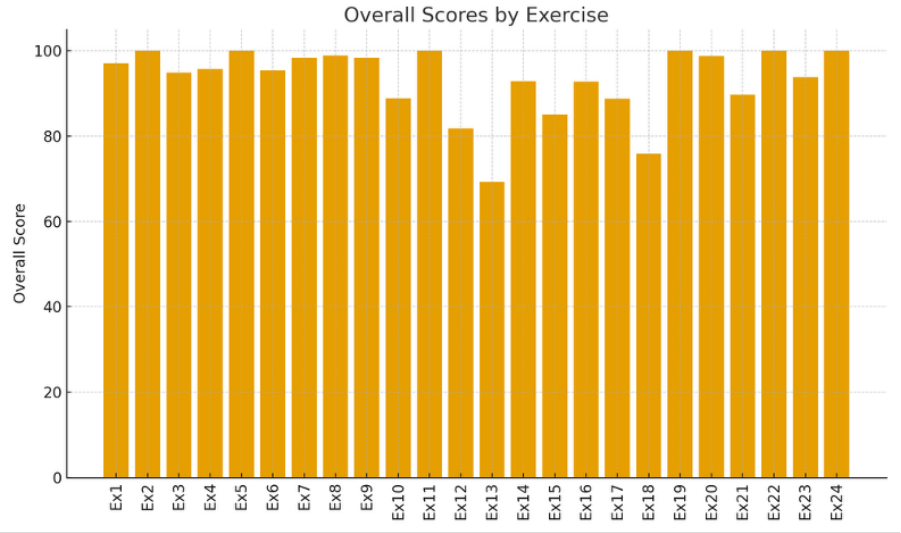

Figure 4.1. Overall scores by exercise.

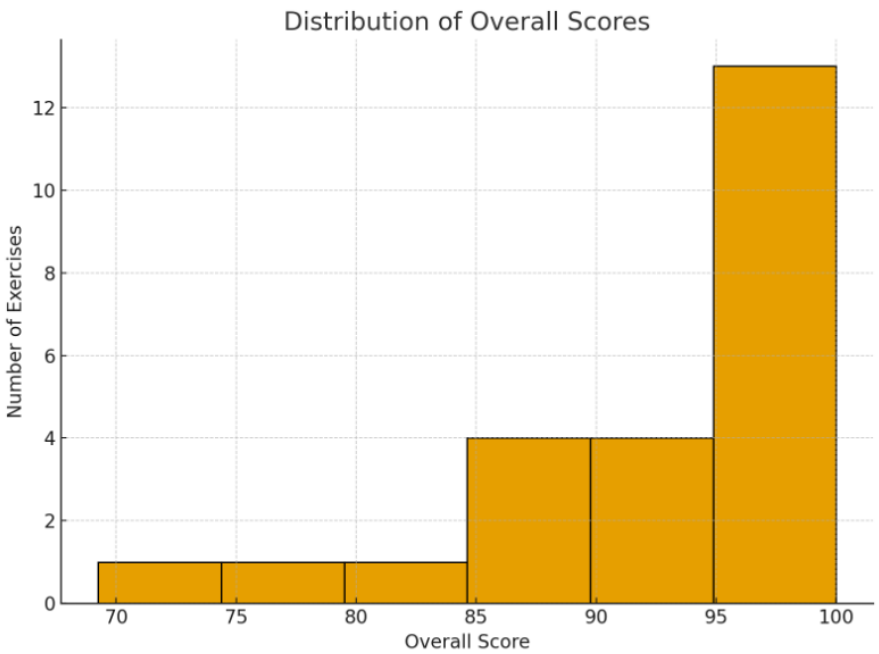

Figure 4.2. Distribution of overall scores.

Response to RQ1: The analysis shows that most BPMN exercises achieved very high overall scores, with 17 out of 24 above 90 and six reaching perfect results. Only a small minority of exercises fell below 85, with Ex13 representing the weakest case (69.25). The distribution is thus dominated by a compact cluster of strong performances, separated from a few outliers that displayed significant weaknesses.

4.2 RQ2: What are the scores obtained for each evaluation dimension by automated analysis of BPMN exercises?

4.2.1 Dimension-Level Results

The performance results can also be examined at the level of the eight evaluation dimensions. Table 4.3 reports the average score, standard deviation, minimum, and maximum values for each dimension, while Figure 4.4 provides a graphical comparison of mean scores together with their standard deviations.
The results show that most dimensions achieved very high values, with averages consistently above 90. Traceability stands out as the most stable dimension, with a mean of 99.72, a very small standard deviation of 1.36, and a minimum as high as 93.33, indicating almost perfect performance across all exercises. Similarly, Ranking (mean = 99.03) and Modifiability (mean = 98.33) demonstrate high levels of accuracy, with only minor variations among exercises. Verificability also performs strongly (mean = 97.71), though with slightly greater variability than Traceability.
By contrast, dimensions such as Consistency (mean = 90.42, std dev = 11.37) and Completeness (mean = 92.78, std dev = 10.50) display wider ranges, with minimum scores falling below 62, highlighting that some exercises posed greater challenges in these areas. Correctness (mean = 92.83) shows a similar pattern, with relatively high overall performance but still some variability across cases. Unambiguousness also records a solid mean of 94.90, though its minimum score of 62.50 reflects notable difficulties in specific exercises.
Taken together, these findings indicate that while several dimensions, particularly Traceability, Ranking, and Modifiability, were handled with near-perfect consistency, others such as Consistency, Completeness, and Correctness revealed a wider spread of results, signaling areas where the model's performance was less uniform. The graphical summary in Figure 4.4 reinforces these observations by showing both the consistently high averages and the relative variability between dimensions.

	CV	media	st dev	min	max	
Completeness	0,1131379	92,7777778	10,496683	61,6666667		100
Consistency	0,12578212	90,4166667	11,3728	58		100
Correctness	0,10887289	92,8292888	10,1065931	62,8571429		100
Modifiability	0,0647819	98,3333333	6,37022057	70		100
Ranking	0,02890343	99,0277778	2,8622424	90		100
Traceability	0,01364618	99,7222222	1,36082763	93,3333333		100
Unambiguousnes	0,10407609	94,8958333	9,87638726	62,5		100
Verificability	0,04766906	97,7083333	4,65766471	85		100

Figure 4.3. Summary of Scores by Dimension

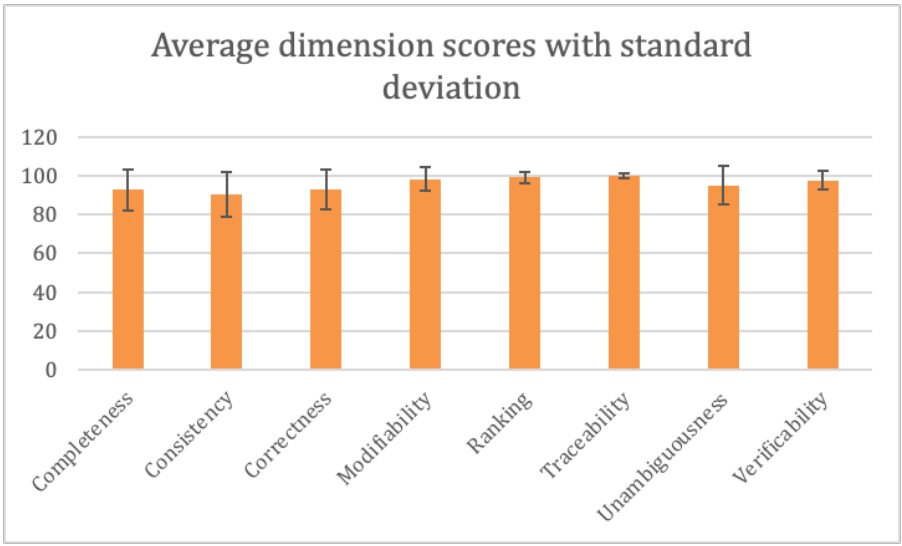

Figure 4.4. Average dimension scores with standard deviation.

4.2.2 Dimension Stability

Beyond average performance, it is also important to assess the relative stability of each dimension across the 24 exercises. To this end, the coefficient of variation (CV) was calculated for every dimension, defined as the ratio between the standard deviation and the mean. A lower CV indicates more stable results across exercises, while higher values reflect greater variability. The outcomes are summarized graphically in Figure 4.5. The results show that Traceability is by far the most stable dimension, with a CV close to zero. This finding confirms that the model consistently achieved near-perfect results in this area, regardless of the exercise considered. Ranking and Modifiability also display very low coefficients of variation, indicating that these dimensions were handled reliably and with minimal fluctuation. Verificability presents similarly strong stability, albeit with slightly higher dispersion than the previous three. By contrast, dimensions such as Consistency and Completeness show the highest coefficients of variation, both exceeding 0.10. This indicates that performance in these areas was much more uneven: while some exercises reached excellent results, others showed substantial weaknesses, producing a wide gap between best and worst cases. Correctness and Unambiguousness occupy an intermediate position: although their average performance was solid, the higher CV values reveal a tendency for results to fluctuate more significantly from one exercise to another. Overall, Figure 4.5 highlights a clear distinction between highly stable dimensions, where the model's performance was uniform and predictable, and more variable dimensions, where results were inconsistent and depended heavily on the specific exercise. This analysis complements the average dimension scores by showing not only how well the model performed on average, but also how reliably it reproduced that performance across different contexts.

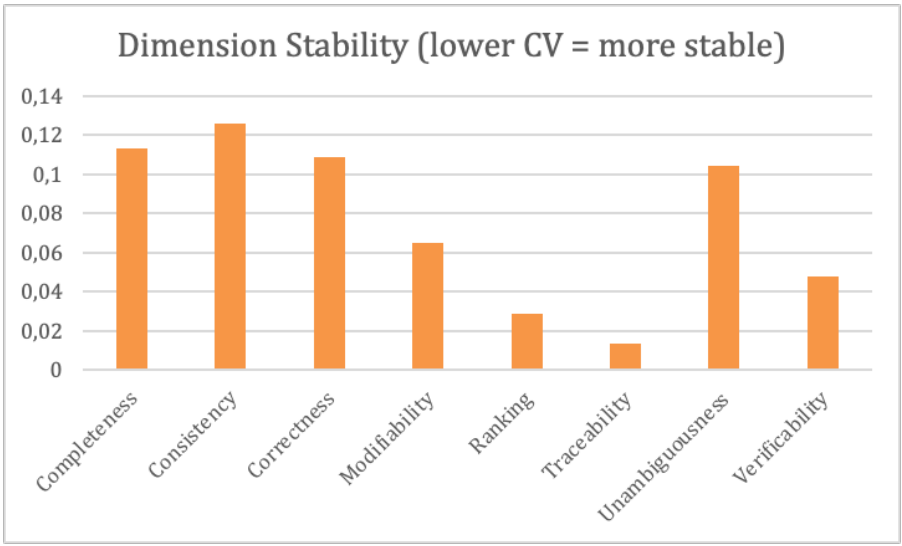

Figure 4.5. Coefficient of variation of dimension scores. Lower values indicate greater stability.

Response to RQ2: Results at the dimension level confirm that while *Traceability*, *Ranking*, and *Modifiability* performed almost perfectly with minimal variability, dimensions such as *Correctness*, *Completeness*, and *Consistency* showed wider dispersion. These latter dimensions therefore represent the most challenging areas for automated analysis and highlight where the evaluation framework identifies significant differences across exercises.

4.3 RQ3: What is the correlation between evaluation dimensions?

4.3.1 Heatmap of Dimension Scores by Exercise

The heatmap in Figure 4.6 visualizes the distribution of normalized scores across all dimensions and exercises, allowing for an immediate comparison of performance patterns both within individual exercises and across the dataset as a whole. The representation confirms that the vast majority of values are concentrated at the top of the scale, with dark shades indicating consistently high results in most dimensions. Exercises such as Ex2, Ex5, Ex11, Ex19, Ex22, and Ex24 reach the maximum score across all dimensions, highlighting their role as reference cases of uniformly strong performance. Other exercises, like Ex20 and Ex23, also show near-perfect results, with only marginal deviations in a limited number of dimensions.

At the same time, the heatmap makes clear where weaknesses emerged. Lower values appear primarily in Completeness, Consistency, and Correctness, which are the dimensions that most frequently deviate from the otherwise high performance pattern. Notable examples include Ex10 (Completeness at 76.9), Ex12 (Correctness at 61.7 and Completeness at 82.1), and Ex13, which records the lowest values across multiple dimensions

simultaneously, with Completeness at 62.9, Correctness at 71.7, Modifiability at 64, and Unambiguousness at 70. Ex18 also stands out for its weaker performance, particularly in Completeness (79.3), Consistency (62.5), Correctness (75), and Modifiability (58). These cases illustrate how weaknesses in one structural dimension often overlap with weaknesses in others. Conversely, certain dimensions show remarkable stability. Traceability, Ranking, and Verificability remain uniformly high across nearly all exercises, rarely deviating from perfect scores. Even in weaker exercises such as Ex13 and Ex18, these dimensions remain close to the maximum, reinforcing their status as the most robust and least variable aspects of model quality. Modifiability also tends to remain strong overall, though with localized exceptions in the weaker cases. The visualization also reveals heterogeneity within single exercises. Ex13, for instance, exhibits low scores in several dimensions simultaneously, underscoring its position as the weakest overall case. By contrast, exercises like Ex19 and Ex22 maintain dark shades across the entire row, confirming their place among the best-performing exercises. Overall, Figure 4.6 provides a detailed cross-sectional perspective: while the dataset is dominated by consistently high results, dimension-specific weaknesses are clearly localized in Completeness, Consistency, and Correctness, with their recurrence across multiple exercises pointing to structural areas where performance was more fragile.

	Completeness	Consistency	Correctness	Modifiability	Ranking	Traceability	Unambiguous	Verificability
Ex1	100	97,5	100	87,5	100	100	100	100
Ex2	100	100	100	100	100	100	100	100
Ex3	89,1666667	100	96	100	100	100	100	100
Ex4	100	75	100	86	100	92,5	100	100
Ex5	100	100	100	100	100	100	100	100
Ex6	100	100	100	80	100	100	100	100
Ex7	100	100	100	92,5	100	100	100	100
Ex8	100	100	94	100	100	100	100	100
Ex9	100	100	100	94	100	92,5	100	100
Ex10	76,9230769	85	94	100	100	92,5	100	100
Ex11	100	100	100	100	100	100	100	100
Ex12	82,1428571	100	61,6666667	85	100	100	100	100
Ex13	62,8571429	85	71,6666667	64	100	85	70	100
Ex14	91,5384615	87,5	98	88	100	100	100	93,3333333
Ex15	80	85	88	85	90	97,5	90	100
Ex16	93,8461538	100	90	88	100	100	100	100
Ex17	92,8571429	100	81,6666667	82	100	85	100	100
Ex18	79,2857143	62,5	75	58	96,6666667	100	100	100
Ex19	100	100	100	100	100	100	100	100
Ex20	98,5714286	100	96,6666667	100	100	100	100	100
Ex21	85	100	88,3333333	92,5	90	100	100	100
Ex22	100	100	100	100	100	100	100	100
Ex23	95,7142857	100	91,6666667	87,5	100	100	100	100
Ex24	100	100	100	100	100	100	100	100

Figure 4.6. Heatmap of dimension scores.

4.3.2 Correlation Between Dimensions

The correlation matrix in Figure 4.7 highlights how the eight evaluation dimensions covary across exercises.

The analysis reveals strong positive correlations among Completeness, Correctness, Consistency, Modifiability, and Unambiguousness, suggesting that these dimensions are tightly interconnected. When a model performs poorly in one of them, it often shows weaknesses

in the others as well. This cluster effect reflects the fact that these dimensions all relate to structural and semantic rigor, and therefore tend to rise and fall together.
By contrast, Ranking and Verificability exhibit weaker correlations with the other dimensions. This indicates that they capture more independent aspects of model quality, less directly influenced by structural integrity. Traceability, meanwhile, shows consistently high scores across exercises, leaving little room for variation and thus limiting its correlation with other measures.
These results demonstrate that while several dimensions operate as an interdependent block, others provide orthogonal information that adds nuance to the evaluation. The combination of these correlated and independent measures ensures that the framework captures both common structural weaknesses and more specific aspects of BPMN model quality.

	Completeness	Consistency	Correctness	Modifiability	Ranking	Traceability	Unambiguous	Verificability
Completeness	1	0,52593813	0,761699	0,60362694	0,37835346	0,45532632	0,69386771	0,02720452
Consistency	0,52593813	1	0,32409818	0,65699661	0,23978471	0,26631883	0,27354699	0,15950224
Correctness	0,761699	0,32409818	1	0,66634368	0,21921649	0,33596054	0,44287746	-0,1059695
Modifiability	0,60362694	0,65699661	0,66634368	1	0,18884854	0,38611728	0,50811586	0,04526131
Ranking	0,37835346	0,23978471	0,21921649	0,18884854	1	-0,0928579	0,14572413	-0,0723497
Traceability	0,45532632	0,26631883	0,33596054	0,38611728	-0,0928579	1	0,56172798	-0,1047999
Unambiguous	0,69386771	0,27354699	0,44287746	0,50811586	0,14572413	0,56172798	1	-0,0557278
Verificability	0,02720452	0,15950224	-0,1059695	0,04526131	-0,0723497	-0,1047999	-0,0557278	1

Figure 4.7. Correlation matrix of evaluation dimensions.

Response to RQ3: Correlation analysis shows that Completeness, Correctness, Consistency, Modifiability, and Unambiguousness form a highly correlated cluster, suggesting strong interdependence among these dimensions. Conversely, Ranking and Verificability display weaker correlations, confirming that they capture independent aspects of model quality. These findings highlight both overlapping and distinct contributions of the evaluation dimensions.

4.4 RQ4: What is the variability of evaluation dimensions within each exercise?

4.4.1 Variability Across Exercises

An additional perspective on the results can be obtained by analyzing the variability of scores within each exercise. Figure 4.8 presents the standard deviation of the eight dimension scores for every exercise, providing a measure of how balanced or uneven the results were internally. A lower standard deviation indicates that all dimensions scored at a similar level, while higher values reveal that some dimensions performed much better or worse than others within the same exercise. The figure shows that most exercises exhibit relatively low to moderate variability, with standard deviations typically below 10. This suggests that, in general, the model's performance across different dimensions tended to be consistent within the same exercise. However, a few cases stand out with much higher

variability. Ex18 records the largest internal dispersion (standard deviation above 17), confirming that its weaknesses were not confined to a single aspect but spread unevenly across multiple dimensions. Similarly, Ex13, Ex12, and Ex15 display higher-than-average variability, reflecting imbalances where certain dimensions performed acceptably while others fell considerably short.
By contrast, several exercises, such as Ex19, Ex22, and Ex24, present almost negligible variability, indicating uniformly strong results across all dimensions. This aligns with their classification as top-performing cases and confirms that their excellence was comprehensive rather than limited to specific aspects of BPMN quality.
Overall, Figure 4.8 highlights that while the majority of exercises showed balanced performance across dimensions, a subset of more challenging cases exhibited sharp internal contrasts. These findings complement the dimension-level analysis by emphasizing not only which exercises scored higher or lower overall, but also how evenly those results were distributed across the eight evaluation categories.

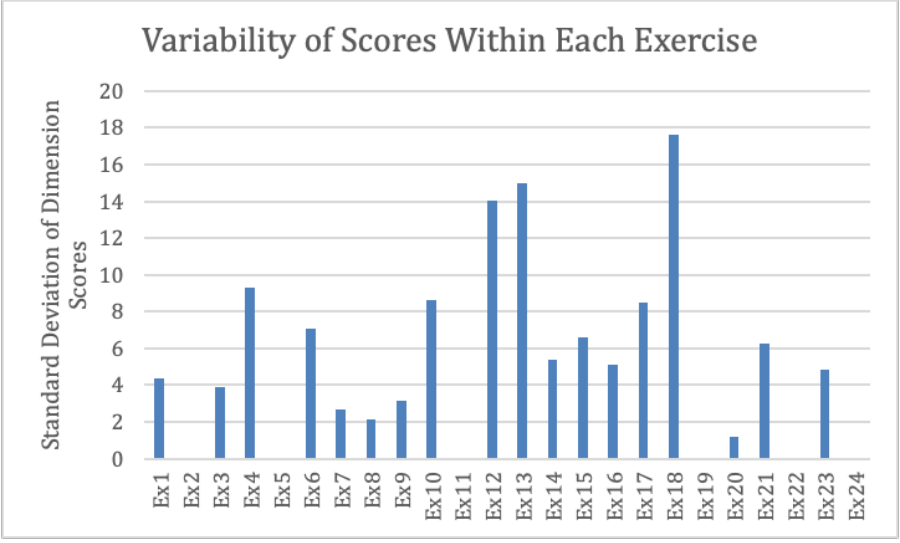

Figure 4.8. Variability of dimension scores within each exercise.

4.4.2 Dimension Weight Impact

The relative contribution of each evaluation dimension to the overall score is illustrated in Figure 4.9, which shows the weighting scheme applied in the calculation of results. This representation clarifies how strongly each dimension influenced the final outcomes, regardless of the raw performance values achieved in individual cases.
The chart makes evident the dominant role of Correctness, which accounts for 40% of the overall score. This dimension therefore had the greatest impact on the results, meaning that even moderate fluctuations in Correctness could significantly raise or lower an exercise's final score. Consistency and Completeness follow, contributing 23% and 20%,

respectively. Together with Correctness, these three dimensions explain more than four-fifths of the overall score, highlighting their central importance in the evaluation framework.
The remaining dimensions carry a much smaller weight. Traceability and Verificability each contribute 4%, while Modifiability, Ranking, and Unambiguousness are weighted at just 3% each. Although these dimensions are not negligible, their limited weight means that weaknesses in these areas had only a modest impact on the final outcome compared to the core trio of Correctness, Consistency, and Completeness.
This weighting distribution underscores the evaluative priorities: the scoring system is designed to emphasize structural and logical rigor over secondary aspects such as modifiability or ranking clarity. As a result, exercises with lower correctness or consistency typically saw a sharp decline in their overall score, even if they performed well in other dimensions. Conversely, exercises that achieved strong results in the three dominant dimensions were able to offset occasional weaknesses in the less heavily weighted categories.

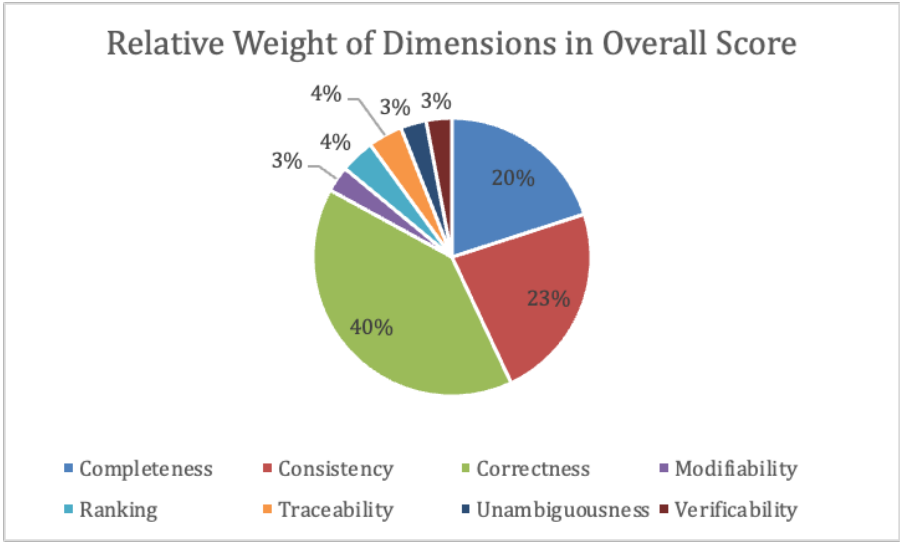

Figure 4.9. Weight of dimensions in overall score calculation.

Response to RQ4: The variability analysis shows that most exercises performed consistently across dimensions, but a few revealed strong internal contrasts, notably Ex12, Ex13, Ex15, and Ex18. The weight distribution confirms that *Correctness*, *Completeness*, and *Consistency* are the most influential dimensions in determining overall results, while dimensions such as *Traceability* and *Ranking* remain uniformly satisfied and less decisive.

Chapter 5

Conclusion

This thesis explored the integration of Large Language Models (LLMs) into the validation of BPMN exercises, with the objective of designing a methodological framework capable of supporting automated assessment in educational contexts. The study was motivated by the limitations of current validation systems, which often rely on predefined reference solutions and require significant manual effort, thereby limiting scalability and flexibility. By introducing LLMs into the process, the research aimed to test whether these models could interpret natural language descriptions of processes, extract the relevant elements in a structured form, and ultimately generate BPMN-compliant models suitable for systematic evaluation. To achieve this, the work developed a two-step pipeline. The first stage focused on the structured extraction of BPMN elements through carefully engineered prompts, gradually refined across multiple iterations to address issues such as conditional logic, pool-lane differentiation, and subprocess representation. The second stage translated these structured outputs into valid BPMN XML files, ensuring importability into standard modeling tools and facilitating a direct comparison with authoritative ground truths. The evaluation was conducted on a curated dataset of 24 heterogeneous exercises, deliberately selected to maximize structural diversity and expose the methodology to a wide range of modeling challenges. The results confirmed the viability of this approach. On average, exercises achieved a high level of accuracy (mean score 93.15), with 17 out of 24 scoring above 90 and six reaching the maximum score of 100. These findings demonstrate that LLMs, when guided by a robust methodological framework, can reliably support BPMN validation. At the same time, the analysis revealed clear differences across evaluation dimensions. Dimensions such as Traceability, Ranking, and Modifiability showed near-perfect stability, confirming that the pipeline consistently preserved logical sequencing and produced models that were auditable and easy to adjust. Conversely, Correctness, Completeness, and Consistency displayed higher variability, with some exercises falling below 70. These dimensions emerged as the most fragile aspects of LLM-based validation, often linked to complex branching structures or implicit textual ambiguities. Correlation analysis provided further insights. Strong interdependencies were observed among Completeness, Correctness, and Consistency, suggesting that weaknesses in one dimension frequently overlapped with weaknesses in the others. By contrast, Ranking and Verificability proved to be more independent, capturing specific

qualities of the models that were less affected by structural challenges. The weighting scheme confirmed the methodological emphasis on structural rigor: Correctness alone accounted for 40% of the overall score, while Completeness and Consistency contributed an additional 43%. This distribution underlines the centrality of these dimensions in determining the quality of BPMN models and explains why performance dips in these areas had such a significant impact on overall results. Beyond numerical outcomes, the study contributes to the broader discussion on the role of LLMs in process modeling. The methodology demonstrated that it is possible to operationalize LLMs in a transparent and reproducible way, avoiding 'black-box' pitfalls by explicitly handling ambiguity and enforcing structural guardrails. In this sense, the proposed pipeline not only achieved high performance but also introduced principles of auditability and comparability that are essential for educational applications.

Future research can build on the present work along several complementary directions. A first and most immediate avenue concerns the development of a dedicated platform that automates the entire pipeline. Instead of relying on the manual submission of prompts, the system could be embedded into a program or web application where the user simply pastes the textual description and directly obtains the corresponding XML file. Such an implementation would not only streamline the process but also enhance reproducibility and accessibility, turning the approach into a practical tool for both research and educational purposes. A second direction involves broadening the set of evaluation indices. While the current framework relies on a fixed number of well-established dimensions, extending the range of indicators could provide a more comprehensive assessment of model quality. Incorporating additional metrics, such as measures of readability, cognitive load, or error-proneness, would allow for a richer understanding of strengths and weaknesses, thereby capturing aspects of process modeling that go beyond structural correctness alone. Finally, future work should explore methods for the automatic optimization of prompts. Rather than relying exclusively on manually designed formulations, techniques such as reinforcement learning, adaptive few-shot prompting, or systematic search strategies could be employed to iteratively refine the instructions given to the model. This would progressively improve stability and performance, reducing the variability inherent in natural language and further consolidating the reliability of the approach. Taken together, these directions emphasize both the practical applicability and the methodological refinement of LLM-based process model evaluation. They suggest a pathway toward tools that are not only more efficient and user-friendly, but also increasingly rigorous in their analytical capabilities.

In conclusion, this thesis demonstrates that LLMs can be successfully integrated into BPMN validation workflows through a carefully designed methodological framework. While challenges remain, particularly with respect to structural correctness and logical consistency, the findings show that LLMs already provide substantial benefits in terms of scalability, flexibility, and reproducibility. These contributions lay the foundation for future work that will further refine the approach and expand its applicability, moving toward a new generation of validation systems that are both technically rigorous and pedagogically impactful.

Appendix A

Repository of Exercises

For completeness, the full set of exercises used in this thesis is provided in two online repositories. These contain the complete material associated with the evaluation and can be accessed through the following links:

- Exercises - Part 1: this folder contains the list of all exercises, including the original text, the reference solution in SVG format, and the reference solution in BPMN format.
- Exercises - Part 2: this folder contains, for each exercise, the input prompt, the solution generated by the language model in textual form, and the corresponding solutions in both BPMN and SVG formats.

Bibliography

[1] Mihai Teodoru. Business process management integration solution in financial sector romania. In *Proceedings of the 5th WSEAS International Conference on Economy and Management Transformation (EMT)*, pages 123–128, Timisoara, Romania, 2009. WSEAS Press.

[2] Xulin Zhao and Ying Zou. A business-process-driven approach for generating e-commerce user interfaces. In *Model Driven Engineering Languages and Systems*. Springer, 2007.

[3] Ana Respício and Dulce Domingos. Reliability of bpmn business processes. *Procedia Computer Science*, 64, 2015.

[4] Alberto De Ramón Fernández, Daniel Ruiz Fernández, and Yolanda Sabuco García. Business process management for optimizing clinical processes: A systematic literature review. *Health Informatics Journal*, 26(2):1305–1320, 2020.

[5] Elvira Rolón, Gabriel Chavira, Jorge Orozco, and Juan Pablo Soto. Towards a framework for evaluating usability of business process models with bpmn in health sector. *Procedia Manufacturing*, 3:5603–5610, 2015.

[6] Emanuel Luciano. Adoption of bpmn in portuguese healthcare institutions: An empirical study. *International Journal of Healthcare Management*, 14(3):789–799, 2021.

[7] Giacomo Garaccione, Riccardo Coppola, and Luca Ardito. Gamifying business process modeling education: A longitudinal study. In *Proceedings of the 28th International Conference on Evaluation and Assessment in Software Engineering (EASE)*, Salerno, Italy, 2024. ACM.

[8] Hajo A. Reijers and Jan Mendling. A study into the factors that influence the understandability of business process models. *IEEE Transactions on Systems, Man, and Cybernetics - Part A: Systems and Humans*, 41(3):449–462, 2011.

[9] Henrik Leopold, Jan Mendling, and Oliver Günther. Learning from quality issues of bpmn models from industry. *IEEE Software*, 33(4):26–33, 2016.

[10] Han van der Aa, Josep Sánchez-Ferreres, Josep Carmona, and Lluís Padró. Aligning textual and model-based process descriptions. *Data & Knowledge Engineering*, 116:101–119, 2018.

[11] Jan Mendling. *Metrics for Process Models: Empirical Foundations of Verification, Error Prediction, and Guidelines for Correctness*. Springer, 2013.

[12] Jan Mendling, Hajo A. Reijers, and Wil M.P. van der Aalst. Seven process modeling guidelines (7pmg). In *Information Systems*, volume 36, pages 467–482, 2012.

[13] Irene Vanderfeesten, Jorge Cardoso, Jan Mendling, Hajo A. Reijers, and Wil M.P. van der Aalst. Quality metrics for business process models. In *Proceedings of the 19th International Conference on Advanced Information Systems Engineering (CAiSE)*, volume 4495 of *Lecture Notes in Computer Science*, pages 177–190. Springer, 2007.