UNIVERSITÀ DI PISA

Department of Informatics

MSc Data Science and Business Informatics

Thesis

Automated Generation of the Dual Partner Skeleton for Business Process Models

Supervisor:

Prof. Roberto Bruni

Submitted by:

Thaharim Khan

Academic Year 2021/2022

ACKNOWLEDGMENTS

I extend my deepest gratitude to my thesis supervisor, Prof. Roberto Bruni, for his invaluable guidance, support, and encouragement throughout the entirety of this research endeavor. His expertise, patience, and unwavering commitment have been instrumental in shaping the direction and quality of this thesis.

I am deeply thankful to the staff and resources provided by the University of Pisa, whose infrastructure and facilities have facilitated the completion of this thesis. Their administrative support and logistical assistance have been indispensable in navigating the challenges of academic research.

My heartfelt thanks go to my family for their unwavering love, encouragement, and understanding throughout my academic journey. Their continuous support and belief in my abilities have been a constant source of motivation and strength.

I am also grateful to my friends and colleagues for their encouragement, camaraderie, and constructive discussions. Their diverse perspectives and intellectual engagement have enriched my research experience and broadened my understanding of the subject matter.

In conclusion, I am indebted to all those who have contributed to the completion of this thesis. Their support and encouragement have been invaluable, and I am profoundly grateful for their assistance in this academic endeavor.

TABLE OF CONTENTS

LIST OF FIGURES

ABSTRACT

This study explores the role of Business Process Model and Notation (BPMN) in enhancing the efficiency and clarity of complex business activities. BPMN provides a standardized visual language for depicting intricate business processes, facilitating communication and understanding among various stakeholders. In addition to conventional BPMN applications, this work introduces a novel concept termed "Dual Partners" to further optimize and coordinate intricate processes involving multiple stakeholders. Dual Partners, represented as complementary processes in a BPMN model, ensure precise alignment of interactions and tasks, reducing confusion, eliminating errors, and expediting decision-making in complex scenarios. The incorporation of Dual Partners into BPMN models requires a thorough understanding of stakeholder relationships and business processes, but the benefits are substantial. This innovative approach enhances the orchestration of complex processes, elevating BPMN to a new level of efficacy and providing enterprises with a critical tool for maintaining a competitive edge in an era of increasing complexity. Additionally, the study includes the development of a library function and its integration with machine learning techniques to further enhance the capabilities of BPMN in managing and optimizing business processes.

Chapter 1

Introduction

1.1 Introduction

Business process modeling (BPM) is a systematic and visual approach to understanding, documenting, analyzing, and improving the workflows and operations within an organization. It is an essential tool that empowers businesses to gain a comprehensive understanding of their processes, identify bottlenecks and inefficiencies, and ultimately enhance productivity and efficiency. BPM is a strategic approach to aligning an organization's business processes with its business goals and objectives. It is not limited to any specific notation or language. Business Process Modeling (BPM) and BPMN (Business Process Model and Notation) are closely related concepts, and BPMN is a specific notation within the broader field of BPM. Business Process Model and Notation (BPMN) is a powerful tool that plays a pivotal role in shaping modern business processes. In the world of business process modeling (BPM), BPMN stands as a universal language that empowers organizations to map, analyze, and optimize their operations. Additionally, the concept of dual partners adds an extra layer of significance, ensuring the effective orchestration of complex processes involving multiple stakeholders. BPMN is a standardized graphical notation used to represent business processes. Its primary importance lies in providing a common framework for business analysts, process designers, and stakeholders to visualize processes in a clear and consistent manner.

Business Process Model and Notation (BPMN) is a Tool for Business process modeling (BPM). BPMN serves as a tool within the broader practice of BPM. It is a graphical notation that helps practitioners create visual representations of business processes. These visual representations, created using BPMN, are a key component of BPM practices. BPMN's standardization ensures that processes are represented consistently and comprehensibly across

different organizations and industries. This standardization is beneficial for achieving clarity and alignment in process documentation and communication. BPMN models are used in the analysis and improvement phases of BPM.

Once a process is modeled using Business Process Model and Notation (BPMN), it can be analyzed for bottlenecks, inefficiencies, and opportunities for enhancement, which is a fundamental aspect of BPM. The models created in BPMN can be integrated into BPM software and automation systems to execute, monitor, and manage processes. This integration is essential for ensuring that the processes operate efficiently and accurately, aligning with BPM's goal of optimizing workflows.

One of the primary reasons organizations need BPMN is to bring clarity to their business processes. Business operations are often complex, involving numerous tasks, interactions, and decision points. BPMN provides a standardized and visual way to represent these processes, making it easier for all stakeholders, from business analysts to process owners, to understand and communicate how these processes work. With BPMN, everyone speaks the same language, reducing misunderstandings and inefficiencies in process design and execution.

In today's digital age, automation plays a significant role in business processes. BPMN models can be seamlessly integrated into workflow management tools and automation systems. This integration streamlines process execution, reduces the likelihood of human errors, and ensures that processes align with the actual operational workflow. BPMN's role in integration and automation is essential for organizations looking to stay competitive and efficient. In many industries, businesses must adhere to strict compliance and regulatory requirements. BPMN is a powerful tool for documenting and demonstrating compliance with these regulations. It provides a clear and structured way to represent processes, making it easier to align with the specific requirements of governing bodies and pass compliance audits. With BPMN, organizations can operate more efficiently, reduce costs, and adapt to the dynamic challenges of the business world, ultimately leading to better outcomes and sustained.

Business Process Model and Notation (BPMN) has become an indispensable tool for organizations seeking to streamline their operations and enhance efficiency. Within this framework, a revolutionary concept emerges—the "Dual Partner". Before delving into the

intricacies of the Dual Partner Generator, it is imperative to comprehend the foundational role of BPMN. As a standardized visual language, BPMN offers a comprehensive representation of complex business processes. It serves as a lingua franca, enabling diverse stakeholders, from analysts to process owners, to grasp and communicate intricate workflows. BPMN's strength lies in its ability to bring clarity to the convoluted. By visually mapping out tasks, interactions, and decision points, BPMN provides a holistic view of business operations. This clarity is the first step toward optimization. The Dual Partner Generator is a conceptual leap forward in BPMN methodology. At its core, it addresses the challenges posed by processes involving multiple stakeholders and intricate dependencies. The generator creates a parallel entity—the Dual Partner—introducing a dynamic and synchronized layer to the traditional linear BPMN approach.

The concept of Dual Partners is grounded in the need for precision and efficiency in complex business processes. As organizations evolve, their operations become more intricate, involving numerous decision points and interactions. Dual Partners serve as strategic counterparts, ensuring that every facet of the process is mirrored accurately.

One of the key advantages of Dual Partners is their ability to minimize errors and reduce confusion. In scenarios where critical decisions must align seamlessly with the intended outcome, Dual Partners act as a fail-safe mechanism. Their presence ensures that the primary process has a synchronized ally, ready to provide support at crucial junctures.

Implementing the Dual Partner Generator involves a nuanced understanding of the underlying business processes. The generator works by analyzing the BPMN diagram, identifying potential points where a Dual Partner can enhance the process. It creates a complementary process that runs parallel to the primary one, stepping in when needed to maintain alignment and accuracy. The Dual Partner Generator introduces a dynamic element to BPMN, acknowledging the fluidity of business operations. As processes evolve, the generator adapts, ensuring that Dual Partners seamlessly integrate with the primary process. This adaptability is crucial in an era where agility and responsiveness are synonymous with success.

The integration of Dual Partners into BPMN diagrams is both an art and a science. It requires a deep understanding of the business context, stakeholder relationships, and potential decision points. The generator examines the BPMN model, identifying areas where

the introduction of a Dual Partner would enhance overall efficiency. In the BPMN diagram, Dual Partners are represented as interconnected entities, intricately linked to the primary process. This visual representation not only aids in understanding the relationships but also serves as a roadmap for implementation. Integrating Dual Partners is a strategic decision, requiring a collaborative effort between business analysts, process owners, and IT specialists.

The introduction of a Dual Partner Generator brings forth a multitude of benefits for organizations embracing BPMN methodology. Firstly, it enhances the robustness of business processes. The synchronized orchestration provided by Dual Partners minimizes the risk of errors, ensuring that the intended outcomes are consistently achieved.

Moreover, the Dual Partner Generator contributes to enhanced decision-making. In scenarios where decisions impact subsequent tasks or interactions, having a Dual Partner ready to validate and support these decisions adds an additional layer of assurance. This, in turn, accelerates the overall pace of the process. From a strategic standpoint, Dual Partners foster collaboration and alignment among stakeholders. In complex processes involving multiple teams or departments, Dual Partners act as unifying elements, ensuring that the various components work in harmony. While the concept of Dual Partners is transformative, its implementation is not without challenges. Organizations must navigate issues such as resource allocation, training, and potential resistance to change. Moreover, the complexity of some processes may require careful consideration to avoid overcomplicating the BPMN diagram.

Additionally, maintaining synchronization between the primary process and Dual Partners requires ongoing monitoring and adjustment. Regular assessments are essential to ensure that the Dual Partner Generator continues to contribute positively to process efficiency. the advent of the Dual Partner Generator within the realm of BPMN diagrams marks a significant evolution in business process management. This innovative concept addresses the growing complexity of modern business operations by introducing a dynamic and synchronized layer to traditional BPMN representations. As organizations strive for efficiency and agility, the Dual Partner Generator emerges as a strategic asset. Its ability to enhance clarity, minimize errors, and foster collaboration positions it at the forefront of BPMN innovation. While challenges exist, the transformative potential of the Dual Partner Generator in optimizing business processes cannot be overstated. Embracing this concept signifies a

commitment to not only understanding but also mastering the intricacies of contemporary business operations. In our thesis work, we are trying to develop a new concept name "Dual Partner". the concept of dual partners adds an extra layer of significance, ensuring the effective orchestration of complex processes involving multiple stakeholders. Dual partner processes in BPMN take the concept of clarity and alignment to the next level. They serve as complementary processes that mirror the original process, ensuring that the interactions and activities are matched accurately.

1.2 Explanations of the problem:

In the intricate landscape of contemporary business processes, where interactions involve multiple stakeholders and complex systems, achieving alignment and consistency is paramount. This article delves into the concept of dual partner processes within the framework of Business Process Model and Notation (BPMN). By exploring the mathematical representation of these processes, we aim to unravel the profound impact they can have on efficiency, accuracy, and alignment in the realm of business process modeling. The Essence of Dual Partner Processes

1.2.1 Alignment and Consistency:

In the realm of complex business processes, ensuring that disparate elements align seamlessly is a persistent challenge. This is where the concept of dual partner processes emerges as a strategic solution. By crafting processes that involve dual partners, organizations can guarantee that interactions and activities on both sides are meticulously matched. This, in turn, mitigates the risks associated with errors or misalignment, fostering a more robust operational environment.

1.2.2 Communication and Collaboration:

Business processes rarely unfold in isolation; rather, they often involve a multitude of stakeholders or systems that interact intricately. The incorporation of dual partner processes serves as a clear and unambiguous representation of these interactions. This clarity, in turn,

facilitates enhanced communication and collaboration among diverse parties involved in the process. The dual partner approach becomes a common language, simplifying the complexity inherent in multifaceted business scenarios.

1.2.3 Documentation and Visualization:

Dual partner processes transcend mere operational strategies; they become invaluable tools for documentation and visualization. These processes provide a tangible representation of how specific interactions or business processes function. Through visualizing the entire process, stakeholders gain a comprehensive understanding of the end-to-end workflow. This not only aids in comprehension but also serves as a foundational document for future reference and improvement initiatives.

1.2.4 The Mathematical Framework: Decoding the Equations:

Now, let's delve into the mathematical representation of dual partner processes. The equation stands as follows:

$$\sum O_P + \sum P_P = S.I \tag{1.1}$$

In this equation: $\sum O_P$ represents the original process, composed of events (e'), gateways (g'), tasks (t'), and flows (f').

$\sum P_P$ represents the partner process, acting as the counterpart to OP, and consists of events (e"), gateways (g"), tasks (t"), and flows (f").

S.I defines the ultimate goal of the dual partner, aspiring to achieve a synchronized interaction between these two processes. Breaking down each component:

$\sum O_P$ comprises events (e'+), gateways (g'+), tasks (t'+), and flows (f'+). For instance, $\sum O_P$ might encompass the initiation of events (e'), decision gateways (g'), execution tasks (t'), and data flows (f').

$\sum P_P$ mirrors $\sum O_P$ but relates to the partner process, hence events (e"), gateways (g"), tasks (t"), and flows (f").

S.I encapsulates the synchronized interaction, represented as events (e+), gateways (g+), tasks (t+), and flows (f+). Mathematically, the equation encapsulates the notion that the

need for a dual partner arises from the desire to ensure:

$$O_P + P_P = OptimalOutcome \tag{1.2}$$

This equation signifies that the dual partner process (P_P) is designed to complement the original process (O_P), ultimately aiming for an optimal outcome in terms of efficiency, accuracy, and alignment between stakeholders or systems.

1.2.5 Practical Implications and Benefits

The utilization of dual partner processes carries substantial benefits for organizations seeking operational excellence. Firstly, it enhances the robustness of business processes by minimizing the risk of errors. The synchronized orchestration provided by dual partners ensures that intended outcomes are consistently achieved. Furthermore, dual partners contribute to enhanced decision-making in scenarios where critical decisions impact subsequent tasks or interactions. The dual partner becomes a strategically, validating and supporting decisions, ultimately accelerating the overall pace of the process. Strategically, dual partners foster collaboration and alignment among stakeholders. In complex processes involving multiple teams or departments, dual partners act as unifying elements, ensuring that various components work in harmony.

1.2.6 Challenges and Considerations

While the concept of dual partners is transformative, its implementation is not without challenges. Organizations must navigate issues such as resource allocation, training, and potential resistance to change. The complexity of some processes may require careful consideration to avoid overcomplicating the BPMN diagram. Maintaining synchronization between the original process and dual partners demands ongoing monitoring and adjustment. Regular assessments are essential to ensure that the dual partner generator continues to contribute positively to process efficiency.

1.2.7 Conclusion

In conclusion, the incorporation of dual partner processes within BPMN diagrams signifies a paradigm shift in business process modeling. These processes address the growing complexity of modern business operations by introducing a dynamic and synchronized layer to traditional BPMN representations. The mathematical representation serves as a powerful tool for understanding the intricacies of dual partner processes. It emphasizes that the combination of the original process and its dual partner should result in an optimal outcome. This outcome encapsulates efficiency, accuracy, and alignment between stakeholders or systems. Embracing the concept of dual partner processes signifies a commitment to not only understanding but also mastering the intricacies of contemporary business operations. As organizations strive for efficiency and agility, the dual partner approach emerges as a strategic asset, unlocking new dimensions of collaboration, alignment, and operational excellence.

Chapter 2

Background

2.1 Introduction

Business Process Model and Notation (BPMN) stands as a pivotal framework for organizations aiming to visualize, analyze, and optimize their business processes. At the heart of BPMN lies a rich set of notations, each carrying specific meanings and functions. This essay delves into the intricacies of BPMN notations, unraveling their significance and providing a comprehensive guide to understanding these visual elements.

2.1.1 Concept of BPMN

BPMN, known as Business Process Modeling and Notation, stands as the universally recognized standard for illustrating business processes, playing a pivotal role in the realm of business process management. By employing BPMN diagrams, diverse stakeholders gain the ability to visually comprehend business processes, facilitating the enhancement of workflows' efficacy and efficiency.

Basic Concepts of BPMN

At its core, BPMN is a standardized visual language that provides organizations with a method to understand their business processes. It offers a common ground for stakeholders, ranging from business analysts to technical developers, to communicate and collaborate effectively.

Elements of BPMN:

BPMN diagrams consist of a variety of elements, each representing a specific aspect of a business process. These elements are categorized into Flow Objects, Connecting Objects, and Swimlanes.

Flow Objects

In BPMN (Business Process Modeling and Notation), flow objects represent various elements that depict the flow of activities within a business process. These flow objects serve as visual markers to illustrate the sequence and interactions between different tasks, events, and gateways. The main types of flow objects in BPMN include.

Events

Events represent something that happens during the course of a process. They can be a start, intermediate, or end event. For instance, a start event can denote the initiation of a process, while an end event signifies its completion.

Activities

Activities represent work that needs to be performed. They can be further classified into tasks and sub-processes. A task is a single unit of work, while a sub-process is a collection of tasks.

Gateways

Gateways control the flow of the process. They represent decision points, forking or merging paths, and conditional flows.

Connecting Objects

There are two connecting objects which is used in BPMN while presenting the BPMN model. Those are Sequence Flow and Message Flow

Sequence Flow

Sequence flow represents the order in which activities are performed. It is depicted by an arrow connecting two elements.

Message Flow

Message flow represents communication between different participants or processes. It is denoted by a dashed line.

Swim lanes

Swim lanes are used to organize and categorize activities within a process. They can be either pools or lanes. Pools represent separate participants or entities, while lanes are subdivisions within a pool.

Advanced BPMN Notations

There are few advance notations which are used for presenting the work with large explanation and figuring out every possible way to represent the overall scenario.

Data Objects

Data objects represent information used or produced during a process.

Data Object (Rectangle with Folded Corner)

Denotes data used or produced in a process.

Data Store (Two Vertical Lines)

Represents the storage of data during a process.

Artifacts

Artifacts provide additional information or clarification about a process.

Group (Rectangle with Dashed Line)

Groups related elements together for organizational purposes.

Annotation (Note Symbol)

Provides additional information or clarification about an element.

BPMN notations form the backbone of effective business process modeling. Understanding these notations is crucial for organizations aiming to streamline their operations, enhance collaboration, and optimize workflows. Whether it's the representation of events, activities, gateways, or swim lanes, each notation carries a specific meaning that contributes to the overall clarity and efficiency of BPMN diagrams. By mastering BPMN notations, organizations can unlock the full potential of this standardized visual language, paving the way for improved communication, better decision-making, and ultimately, operational excellence. As the business landscape continues to evolve, BPMN notations remain a powerful tool for those navigating the complexities of modern business processes.

2.1.2 Tools for BPMN

There are many tools are available both in online and desktop version for implementing the BPMN model. Those tool are useful for generating the scenario in a graphical view. In the dynamic landscape of business process modeling, Python has emerged as a powerful ally, offering a range of libraries tailored to facilitate the implementation of Business Process Model and Notation (BPMN). This essay explores some notable Python libraries dedicated to BPMN, shedding light on their features, applications, and contributions to efficient process management.

Understanding BPMN in Python

Before delving into the Python libraries, let's briefly revisit BPMN. Business Process Model and Notation is a visual language that provides a standardized way to represent and communicate business processes. It employs a set of graphical elements to depict the flow,

events, tasks, and decisions within a process. Utilizing BPMN in Python ensures a common ground for stakeholders, facilitating comprehension and collaboration.

Exploring tools for BPMN

There are several tools are used for implementing BPMN diagram. Those are given below

Camunda BPM

Camunda is an open-source platform that supports BPMN, CMMN (Case Management Model and Notation), and DMN (Decision Model and Notation). Although Camunda is primarily implemented in Java, it offers a Python client, enabling seamless integration with Python applications. Camunda's versatility makes it suitable for various BPM scenarios, from simple workflows to complex business processes.

Features

- Execution of BPMN processes in Python applications.
- Integration with popular Python frameworks.
- Support for external task workers written in Python.

Activiti

Activiti, like Camunda, is a Java-based BPM platform that supports BPMN. Its extensible nature allows integration with Python through REST APIs. Activiti provides a range of features for process modeling, execution, and monitoring.

Features

- RESTful APIs for interaction with Python applications.
- Flexible process modeling and execution.
- Support for cloud deployment, enhancing scalability.

Zeebe-Python

Zeebe is a workflow engine designed for microservices orchestration. Zeebe-Python is a Python client for Zeebe, enabling Python developers to participate in Zeebe-based BPMN workflows.

Features

- Python API for interacting with Zeebe.
- Integration with microservices architectures.
- Scalable and fault-tolerant workflow execution.

Zeebe-Python

Flowable

Flowable is an open-source BPM platform with extensive support for BPMN. While it is primarily implemented in Java, it offers REST APIs that allow seamless integration with Python applications.

Features

- RESTful APIs for BPMN execution.
- Robust process modeling and execution capabilities.
- Support for decision tables and case management.

Signavio Workflow Accelerator

Signavio Workflow Accelerator is a cloud-based workflow automation tool that supports BPMN. While it is not a traditional Python library, it offers REST APIs, making it accessible for integration with Python-based applications.

Features

- RESTful APIs for BPMN execution.
- Cloud-based BPMN modeling and execution.
- Collaboration features for distributed teams.

Cawemo

Cawemo is a web-based tool for collaborative BPMN modeling. Although it doesn't directly provide a Python library, it offers RESTful APIs that allow Python developers to interact with BPMN models created using Cawemo.

Features

- RESTful APIs for BPMN execution.
- Collaborative BPMN modeling in a web-based environment.
- Version control and change tracking for BPMN models.

BPMN4Tosca

BPMN4Tosca is a library designed to support the BPMN and TOSCA (Topology and Orchestration Specification for Cloud Applications) standards. It enables Python developers to work with BPMN and TOSCA models efficiently.

Features

- Python API for BPMN and TOSCA modeling.
- Support for orchestrating cloud applications.
- Extensibility for custom requirements.
- Practical Applications and Use Cases

There are several benefits for these tools which is immensely helped while building up the whole concept. Below some benefits are enlisted

Workflow Automation

Python libraries for BPMN empower organizations to automate workflows seamlessly. This is particularly valuable in scenarios where repetitive tasks can be streamlined, reducing manual effort and minimizing errors.

Microservices Orchestration

Libraries like Zeebe-Python and Camunda BPM support microservices orchestration. Python developers can leverage these tools to design and execute complex business processes in a microservices architecture.

Collaborative Modeling

Platforms such as Cawemo and BPMN.io, with their collaborative features, facilitate team-based BPMN modeling. Python applications can integrate with these platforms to incorporate collaborative workflows.

Cloud-Based Process Management

Cloud-based tools like Signavio Workflow Accelerator offer the flexibility of BPMN modeling and execution in a cloud environment. Python applications can interact with these tools using RESTful APIs.

Challenges and Considerations

While Python libraries for BPMN bring numerous advantages, certain challenges should be considered:

- Integration Complexity: Integration with BPMN tools often requires understanding and utilizing RESTful APIs. This can pose challenges for developers unfamiliar with these integration mechanisms.

- Learning Curve: Some BPMN platforms, though offering Python support, may have a learning curve, especially if developers are new to BPMN concepts.
- Scalability: In scenarios with a high volume of workflows or complex orchestration requirements, ensuring the scalability of the chosen BPMN library is essential.

The realm of BPMN in Python is dynamic and expansive, offering a variety of libraries and tools to cater to diverse business process modeling needs. Whether it's the microservices orchestration capabilities of Zeebe-Python, the collaboration features of Cawemo, or the extensive functionality of Camunda BPM, Python developers have a rich toolkit to streamline and optimize business processes. As organizations increasingly recognize the value of efficient process management, Python libraries for BPMN stand as essential assets, providing the means to translate complex workflows into manageable, automated, and collaborative processes. As the landscape continues to evolve, these libraries will play a pivotal role in shaping the future of business process modeling and execution in Python.

2.1.3 Libraries used for BPMN in python

In the realm of Business Process Model and Notation (BPMN), Python stands as a powerful ally, offering a diverse range of libraries that simplify BPMN workflows. From visualizing BPMN diagrams to programmatically interacting with them, Python provides a robust ecosystem. This article explores key BPMN libraries, including ElementTree, bpmn.io, bpmn-python, BPMN-RPA, and ProcessPiper. We'll delve into each library's functionalities, provide examples, and demonstrate how to parse them effectively.

Understanding BPMN with Python libraries

Before we dive into the libraries, let's briefly revisit the fundamental concepts of BPMN in Python. Business Process Model and Notation is a visual language that standardizes the representation of business processes. In Python, leveraging BPMN allows for seamless communication and collaboration among stakeholders. Here we Explore some important BPMN Libraries with Examples

- **ElementTree:** ElementTree, a built-in Python library, is a versatile tool for XML parsing. While not BPMN-specific, it is widely used for handling BPMN files due to its simplicity. Let's explore how to parse a BPMN file using ElementTree.

```
import xml.etree.ElementTree as ET
# Load BPMN file
tree = ET.parse('sample.bpmn')
root = tree.getroot()
# Access BPMN elements
for element in root.iter():
print(element.tag, element.attrib)
```

This example demonstrates how to load a BPMN file and iterate through its elements using ElementTree.

- **bpmn.io:** Description: bpmn.io is a JavaScript toolkit for BPMN visualization. Although not Python-native, it can be integrated into Python web applications. Let's consider an example using Flask..

```
from flask import Flask, render_template
app = Flask(__name__)
@app.route('/')
def index():
 return render_template('bpmn_viewer.html')
if __name__ == '__main__':
    app.run(debug=True)
```

This example uses Flask to render an HTML page containing the bpmn.io viewer, creating a simple web-based BPMN visualization.

- **bpmn-python:** Description: bpmn-python is a Python library specifically designed for BPMN 2.0. It offers capabilities for creating, modifying, and analyzing BPMN diagrams programmatically. Let's explore how to create a simple BPMN process using bpmn-python.

```
from bpmn_python.bpmn_diagram import BpmnDiagram
# Create a BPMN diagram
diagram = BpmnDiagram()
# Add a process
process = diagram.add_process(id='process_1', name='Sample Process')
# Add a task
task = process.add_task(id='task_1', name='Sample Task')
# Connect task to the process
process.add_sequence_flow(start_id=process.start_event.id, end_id=task.id)
process.add_sequence_flow(start_id=task.id, end_id=process.end_event.id)
# Export BPMN XML
bpmn_xml = diagram.export_xml()
print(bpmn_xml)
```

This example demonstrates how to use bpmn-python to create a simple BPMN process with a task and export it to BPMN XML.

- **BPMN-RPA:** BPMN-RPA focuses on Robotic Process Automation (RPA) within BPMN processes. It integrates RPA capabilities into BPMN workflows. Let's consider an example of how to automate a task using BPMN-RPA.

```
from bpmn_rpa import BpmnRpa
# Create a BPMN-RPA instance
bpmn_rpa = BpmnRpa()
# Define an RPA task
rpa_task = bpmn_rpa.add_rpa_task
(name='Automated Task', script='python_script.py')
# Add the RPA task to a BPMN process
bpmn_rpa.add_to_process(rpa_task)


# Export BPMN-RPA XML
bpmn_rpa_xml = bpmn_rpa.export_xml()
print(bpmn_rpa_xml)
```

This example showcases how to use BPMN-RPA to integrate a Python script into a BPMN process for automation.

- **ProcessPiper:** ProcessPiper is a Python library designed for BPMN workflows, offering functionalities for creating, modifying, and analyzing BPMN diagrams. Let's explore how to parse a BPMN file using ProcessPiper. .

```
from processpiper.bpmn import BPMNFile
# Load BPMN file
bpmn_file = BPMNFile('sample.bpmn')
# Access BPMN elements
for element in bpmn_file.processes[0].tasks:
print (element.id, element.name)
```

This example demonstrates how to load a BPMN file and iterate through its elements using Process Piper.

There are several Practical Applications and Use Cases of these mentioned libraries given follow.

- **Workflow Automation:** Python libraries such as bpmn-python and BPMN-RPA empower organizations to automate workflows seamlessly. For instance, BPMN-RPA allows the integration of Python scripts into BPMN processes, enhancing automation and efficiency.
- **Microservices Orchestration:** Libraries like bpmn-python and Zeebe-Python support microservices orchestration within BPMN workflows. Developers can leverage these tools to design and execute complex business processes in a microservices architecture.
- **Collaborative Modeling:** Platforms such as bpmn.io and Cawemo, with their collaborative features, facilitate team-based BPMN modeling. Python applications can integrate with these platforms to incorporate collaborative workflows.
- **Cloud-Based Process Management:** Cloud-based tools like Signavio Workflow Accelerator offer the flexibility of BPMN modeling and execution in a cloud environment. Python applications can interact with these tools using RESTful APIs.

Challenges and Considerations

While these Python libraries offer numerous advantages, certain challenges should be considered:

- **Integration Complexity:**Integrating with BPMN tools often requires understanding and utilizing RESTful APIs. This can pose challenges for developers unfamiliar with these integration mechanisms.

- **Learning Curve:** Some BPMN platforms, though offering Python support, may have a learning curve, especially for developers new to BPMN concepts.

- **Scalability:** Ensuring the scalability of the chosen BPMN library is essential, particularly in scenarios with a high volume of workflows or complex orchestration requirements.

2.1.4 Conclusion

In conclusion, the world of BPMN in Python is diverse and dynamic, providing a rich toolkit for developers to streamline and optimize business processes. Whether it's the visualizations offered by bpmn.io, the programmatic capabilities of bpmn-python, or the automation features of BPMN-RPA, Python developers have a range of libraries at their disposal. As organizations increasingly recognize the value of efficient process management, these Python libraries for BPMN become indispensable assets. They offer the means to translate complex workflows into manageable, automated, and collaborative processes. As the landscape continues to evolve, these libraries will play a pivotal role in shaping the future of business process modeling and execution in Python.

Chapter 3

Design

3.0.1 Introduction

In the intricate landscape of business process modeling, the concept of dual partners emerges as a powerful paradigm, allowing organizations to represent and accommodate multiple perspectives within the same BPMN diagram. This article embarks on an exploration of the notations and transformations crucial for crafting a dual partner in a BPMN workflow. We will delve into the subtleties of altering gateways, tasks, events, message flows, and data objects, providing detailed descriptions and examples to illuminate the path towards enhanced collaboration and clarity.

3.0.2 Understanding Dual Partner Transformations in BPMN

- **Gateways:** gateways in BPMN act as decision points, determining the flow of a process based on conditions or events. When transforming for a dual partner, exclusive gateways can metamorphose into event-based gateways and vice versa, offering a nuanced way to represent alternative decision-making perspectives.

  Example:

  – Original Process: $A \rightarrow (ExclusiveGateway) \rightarrow B$

  – Dual Partner: $A \rightarrow (EventBasedGateway) \rightarrow B$

  In this transformation, the dual partner introduces an event-based gateway, allowing decisions to be based on events rather than conditions.

Additional Insights:

- Event-based gateways open avenues for modeling processes where decisions hinge on occurrences rather than predefined conditions.
- Dual partner gateways provide a flexible structure, enabling dynamic adaptation to varying process scenarios

- **Tasks:** Tasks in BPMN represent work units or activities within a process. In the context of dual partners, tasks can undergo transformations to align with the alternative perspective. For instance, a send task in the original process can transition into a receive task in the dual partner, fostering a symbiotic relationship.

  Example:

  - Original Process: $A \rightarrow (sendtask) \rightarrow B$
  - Dual Partner: $A \rightarrow (receivetask) \rightarrow B$

  This transformation mirrors the communication aspect, ensuring that interactions in the dual partner align with the original process. .

  Additional Insights:

  - Dual partner tasks facilitate a reciprocal flow of information, emphasizing the bidirectional nature of collaborative processes.
  - The transformation from send to receive tasks underscores the complementary nature of dual partners.

- **Events:** Description: Events in BPMN signify occurrences or states within a process. Dual partner transformations often involve converting throw events to catch events and vice versa. This alteration ensures that both perspectives synchronize their interactions effectively.

Example:

- Original Process: $A \rightarrow (throwevent) \rightarrow B$
- Dual Partner: $A \rightarrow (catchevent) \rightarrow B$

This transformation aligns events in both partners, emphasizing a shared understanding of process occurrences. .

Additional Insights:

- Dual partner event transformations enhance synchronization, promoting a coherent and coordinated workflow.
- Catch events in the dual partner mirror throw events in the original process, creating a harmonious representation of shared states.

- **Message Flow:** Message flows in BPMN depict communication between two entities. Dual partner transformations involve adjusting the direction of message flows to ensure a seamless exchange of information. The sender in the original process corresponds to the receiver in the dual partner, and vice versa.

Example:

- Original Process: $A \rightarrow (MessageFlow) \rightarrow B$
- Dual Partner: $B \leftarrow (MessageFlow) \leftarrow A$

This transformation aligns the communication channels, establishing a reciprocal relationship between the original process and its dual partner.

Additional Insights:

- Message flow adjustments enhance the clarity of communication channels, fostering a comprehensive understanding between dual partners.

  - Dual partner message flows contribute to a bidirectional exchange of information, reinforcing collaborative dynamics.

- **Data Objects:** Data objects in BPMN represent information exchanged between process elements. Dual partner transformations involve mirroring or adapting data objects to maintain consistency in the flow of information between partners.

  Example:

  - Original Process: $A \rightarrow (DataObject) \rightarrow B$
  - Dual Partner: $A \leftarrow (MirroredDataObject) \leftarrow B$

  This transformation ensures that data objects in the dual partner mirror or complement those in the original process, fostering a cohesive information flow. .

  Additional Insights:

  - Consistent data object transformations contribute to a seamless exchange of information, promoting collaboration and coherence.
  - Mirrored data objects emphasize the reciprocal nature of information sharing between dual partners.

Practical Applications and Use Cases

- **Improved Collaboration:** Dual partner transformations in BPMN lay the foundation for improved collaboration by providing a clear and comprehensive representation of how two perspectives align within a shared workflow.

- **Enhanced Clarity:**The meticulous adjustments to gateways, tasks, events, message flows, and data objects contribute to enhanced clarity, ensuring that both dual partners share a common understanding of the collaborative process.

- **Streamlined Communication:**The meticulous alignment of notations in dual partner transformations facilitates streamlined communication between different entities involved in a business process. This leads to a more efficient and coherent workflow.

- **Flexible Process Design:**Dual partner notations offer flexibility in process design, allowing organizations to adapt and model their workflows based on the diverse perspectives of stakeholders or systems. This adaptability contributes to the agility of business processes.

Challenges and Considerations

While the benefits of dual partner transformations are significant, certain challenges should be considered:

- **Consistency:** Ensuring consistency between the original process and its dual partner is paramount. Any discrepancies may lead to misunderstandings and inefficiencies in the collaborative workflow.

- **Documentation:** Thorough documentation of dual partner transformations is essential for maintaining clarity and facilitating communication among stakeholders. A comprehensive record aids in understanding the rationale behind each transformation.

- **Validation:** Rigorous validation of the dual partner process is crucial to identify any errors or misalignments early in the design phase. Automated validation tools can enhance the accuracy and reliability of the dual partner BPMN diagram.

The intricate dance of dual partners in BPMN transforms traditional process modeling into a dynamic and collaborative endeavor. The notations and transformations explored in this article offer a comprehensive toolkit for organizations seeking to represent multiple perspectives within the same workflow. By understanding and implementing the nuances of gateway adjustments, task realignments, event transformations, message flow adaptations, and data object mirroring, businesses can craft BPMN diagrams that foster enhanced collaboration, clarity, and communication. As organizations continue to navigate the complex

terrain of business processes, the utilization of dual partner notations stands as a valuable strategy for optimizing workflows and accommodating diverse stakeholder viewpoints.

3.0.3 List of requisites

Dual partner in a BPMN (Business Process Model and Notation) diagram involves a set of requisites to ensure a seamless and effective representation of multiple perspectives within the same workflow. Here is a comprehensive list of requisites for the creation of a dual partner in BPMN:

- Clear Understanding of Business Process:A thorough comprehension of the original business process is essential before attempting to create a dual partner. Understanding the sequence of activities, decision points, and communication channels is foundational.

- Dual Partner Objectives and Alignment: Define the specific objectives of the dual partner. Whether it's mirroring the original process or introducing alternative decision-making perspectives, clarity on the purpose of the dual partner is essential.

- Notation Mapping: Establish a mapping of notations between the original process and its dual partner. This includes mapping gateways, tasks, events, message flows, and data objects to ensure consistent representation and alignment.

- Gateway Transformation Rules:Define transformation rules for gateways. Determine how exclusive gateways in the original process will be transformed into event-based gateways or vice versa. Establish criteria for making decisions in each perspective.

- Task Transformation Guidelines:Develop guidelines for transforming tasks between the original process and its dual partner. Decide how send tasks become receive tasks and vice versa, ensuring that the communication aspects align.

- Event Transformation Criteria:Clearly define criteria for transforming throw events into catch events and vice versa. Specify the conditions under which events will be mirrored or altered to maintain synchronization between dual partners.

- Message Flow Directional Adjustments:Establish rules for adjusting the direction of message flows. Ensure that the sender in the original process corresponds to the receiver in the dual partner and vice versa. Maintain consistency in communication channels.

- Data Object Mapping Strategy:Define a strategy for mapping data objects between the original process and its dual partner. Determine how data objects will be mirrored or adapted to maintain a cohesive flow of information between partners.

- Consistency Checks: Implement consistency checks to ensure that the dual partner representations align with the original process. Regularly validate the BPMN diagram for any discrepancies or misalignments.

- Documentation Practices:Adopt thorough documentation practices for dual partner transformations. Document the rationale behind each transformation, making it easier for stakeholders to understand the decisions made during the process modeling.

- Validation Mechanisms: Utilize validation mechanisms, including automated tools if available, to validate the dual partner BPMN diagram. This helps identify and rectify errors or misalignments early in the design phase.

- Version Control and Change Management: Implement version control and change management practices to track revisions and updates to the dual partner BPMN diagram. This ensures transparency and traceability in the evolution of the business process models.

By adhering to these requisites, organizations can navigate the complexities of dual partner creation in BPMN, fostering a collaborative and comprehensive representation of varied perspectives within a shared workflow.

The successful creation and implementation of dual partner modeling in BPMN involve certain software and hardware requisites to ensure a smooth and efficient workflow. Below is a comprehensive list of requisites for both software and hardware aspects:

Software Requisites:

- **BPMN Modeling Tool:** A dedicated BPMN modeling tool is crucial for creating, editing, and visualizing BPMN diagrams. Choose a tool that supports the BPMN 2.0 standard and provides a user-friendly interface for efficient dual partner modeling.

- **Version Control System:** Implement a version control system to track changes, revisions, and updates to BPMN diagrams. This ensures a systematic approach to managing dual partner transformations, facilitating collaboration and traceability.

- **Validation and Testing Tools:** Utilize BPMN validation and testing tools to automatically check for consistency, correctness, and adherence to BPMN standards. These tools help identify errors or misalignments in dual partner diagrams during the modeling phase.

- **Collaboration Platform:** Implement collaboration platform or project management tool that allows team members and stakeholders to collaborate on dual partner modeling. Features such as real-time editing, commenting, and version history enhance collaboration.

- **Documentation Software:** Use documentation software to create detailed records of dual partner transformations. Comprehensive documentation aids in understanding the rationale behind each notation and transformation, facilitating communication among team members.

- **Integration with Other Systems:**Ensure that the BPMN modeling tool can integrate seamlessly with other systems, databases, or workflow management tools that the business process interacts with. This integration enhances the holistic representation of dual partners.

- **Security and Access Control:**Implement security measures and access controls within the BPMN modeling tool and collaboration platform. This ensures that only authorized individuals have access to sensitive business process information.

- **Training Resources:**Provide training resources or tutorials for team members to familiarize themselves with the selected BPMN modeling tool and associated software. This ensures proficiency in utilizing the tools for dual partner modeling.

- **Change Management System:**Establish a change management system that allows for systematic approval, tracking, and implementation of changes to dual partner BPMN diagrams. This ensures a controlled and documented process for modifications.

Hardware Requisites:

- **Computing Devices:** Ensure that team members have access to reliable computing devices (desktops, laptops, or tablets) capable of running the chosen BPMN modeling tool and associated software.

- **Sufficient RAM and Processing Power:** Adequate RAM and processing power are essential for smooth operation of BPMN modeling tools, especially when working with complex diagrams. Ensure that the hardware meets or exceeds the recommended specifications of the selected tools.

- **High-Resolution Displays:** High-resolution displays contribute to a comfortable and visually clear workspace, especially when working with detailed BPMN diagrams. Consider monitors with ample screen real estate for improved productivity.

- **Internet Connectivity:** Reliable internet connectivity is crucial for accessing collaborative platforms, cloud-based BPMN modeling tools, and resources required for dual partner modeling. A stable connection ensures seamless collaboration and data synchronization.

- **Backup and Storage Solutions:** Implement backup and storage solutions to safeguard dual partner BPMN diagrams and associated documentation. Regular backups prevent data loss and provide a means of recovering information in the event of unforeseen issues.

- **Peripheral Devices:**Consider peripheral devices such as ergonomic keyboards, mice, and drawing tablets for enhanced user experience, especially when intricate details need to be captured in BPMN diagrams.

- **Printing and Plotting Devices:** In scenarios where physical copies of BPMN diagrams are required, access to reliable printing and plotting devices is necessary. Ensure that these devices support the size and quality requirements of the diagrams.

By addressing these software and hardware requisites, organizations can establish a robust foundation for the creation, collaboration, and implementation of dual partner modeling in BPMN. These requisites contribute to an efficient, secure, and collaborative environment for representing multiple perspectives within shared business processes.

3.0.4 Design of Choice

The world of Business Process Model and Notation (BPMN) is dynamic, offering a myriad of tools and libraries to streamline process modeling. Python, being a versatile programming language, has witnessed the emergence of various libraries designed to interact with and manipulate BPMN diagrams. However, when it comes to the intricacies of dual partner modeling, certain limitations may arise. In this exploration, we delve into why Python libraries like Element Tree, bpmn.io, bpmn-python, BPMN-RPA, and Processpiper might face challenges in the nuanced task of creating dual partners within BPMN diagrams.

Limitation of Element Tree :

Lack of BPMN-Specific Functionality for Dual Partner Modeling Element Tree is a powerful XML manipulation library in Python, often used for parsing and creating XML documents. While it is handy for generic XML tasks, its lack of BPMN-specific functionality poses a challenge for tasks as nuanced as dual partner modeling.Element Tree is primarily focused on generic XML handling and lacks the domain-specific features needed for BPMN modeling.Dual partner modeling involves intricate transformations of BPMN elements, which may go beyond the capabilities of a generic XML library. Ensuring consistency between the

original process and its dual partner requires BPMN-aware functionalities, which Element Tree does not inherently provide.

Limitation of bpmn.io:

Visualization and Interaction Focus, Limited Programmability. bpmn.io is a popular BPMN toolkit that provides a robust solution for BPMN visualization and user interaction. However, its primary emphasis lies in creating visually appealing diagrams and supporting user interactions rather than programmatically manipulating BPMN elements.While bpmn.io excels in rendering BPMN diagrams, its capabilities for programmatic modifications might be limited.Dual partner modeling involves intricate transformations and might require more flexibility in terms of modifying the underlying BPMN elements.The toolkit may not provide the level of control needed for detailed alterations required in dual partner scenarios.

Limitation of bpmn-python:

Limited Documentation and Community Support. BPMN-Python is a library explicitly designed for working with BPMN diagrams in Python. However, its limitations become apparent in terms of documentation and community support.Insufficient documentation might hinder the understanding and utilization of specific features required for dual partner modeling.Community support plays a crucial role in addressing issues and evolving the library; limited community engagement might result in slower responses to queries or issues. Dual partner modeling often requires a collaborative and well-supported environment, which might be challenging with limited community backing.

Limitation of BPMN-RPA:

Focus on Robotic Process Automation (RPA) Specifics BPMN-RPA is designed to bridge the gap between BPMN and RPA, providing features that streamline the integration of business processes with robotic automation. However, its specific focus on RPA may limit its suitability for dual partner modeling. The emphasis on RPA-centric functionalities might lead to less flexibility in terms of accommodating the intricacies of dual partner transfor-

mations.While suitable for certain automation scenarios, the library might not provide the depth of features required for comprehensive dual partner modeling.

Limitation of Processpiper:

Potential Overhead for Simpler Use Cases.Processpiper is a Python library focused on BPMN 2.0. While it offers a range of features for BPMN manipulation, its comprehensive nature might introduce unnecessary complexity for simpler use cases like dual partner modeling. For straightforward tasks like dual partner modeling, a library with extensive features might introduce an overhead in terms of implementation and learning curve.The versatility of Processpiper might be more suitable for complex BPMN scenarios, potentially making it less streamlined for simpler dual partner transformations.

While the mentioned Python libraries serve various BPMN-related purposes, they may face limitations when specifically tasked with dual partner modeling. The intricacies of representing multiple perspectives within the same workflow demand a specialized set of functionalities and a high degree of flexibility, which might not be the primary focus of these libraries. Organizations and developers venturing into dual partner modeling in BPMN may need to consider a combination of BPMN-aware libraries, tools with strong programmability features, and potentially custom implementations to meet the specific requirements of dual partner scenarios. As the field of BPMN evolves, it's crucial to stay abreast of new developments and emerging libraries that might better cater to the nuanced needs of dual partner modeling in Python.

3.0.5 Possible alternative

- **Purpose and Core Objectives:** At its core, "bpmn-transform" is designed to streamline the creation of dual partners in BPMN diagrams. The library focuses on automating the intricate task of transforming and mirroring BPMN elements to represent multiple perspectives within a shared business process. The primary objectives of the library include:
  - Simplifying the process of generating dual partners.

  - Ensuring consistency and alignment between the original process and its dual partners.
  - Providing a flexible and customizable framework for dual partner modeling.

- **Core Functionalities:** The "bpmn-transform" library encompasses a range of functionalities, each strategically crafted to address specific aspects of dual partner generation:
  - **Element Replacement:** The library excels in seamlessly replacing BPMN elements crucial for dual partner modeling. Tasks, gateways, and intermediate events undergo systematic transformations to create synchronized interactions.
  - **Namespace Management:** Efficient handling of namespaces is pivotal in BPMN. "bpmn-transform" intelligently manages namespace prefixes, ensuring a cohesive representation of the modified BPMN diagram.
  - **Sequence Flow Adjustment:** Recognizing the importance of data flow, the library automatically updates sequence flows to reflect the changes introduced during element replacements.
  - **Data Object and Data Store Handling:** "bpmn-transform" offers functionality to remove DataObjectReference and DataStoreReference elements, streamlining the BPMN diagram for dual partner clarity.

3.0.6 Usages and Integration:

- **Installation:** Before diving into its usages, integrating the "bpmn-transform" library into your Python environment is straightforward. Utilize pip, the Python package installer, with the following command: **pip install bpmn-transform**

- **Basic Usage:** Employing "bpmn-transform" in your workflow is a step-by-step process. The following code snippet illustrates a simple use case:

```
from bpmn_transform import process_bpmn_file
if __name__ == "__main__":
```

```
input_file_name = input
("Enter the name of the input BPMN file:")
process_bpmn_file(input_file_name)
print(f"Modified BPMN file saved as{input_file_name.replace('.b
```

- **Input Interaction:** The library begins by prompting the user to enter the name of the input BPMN file. This establishes a user-friendly interaction, allowing seamless integration into various user-driven scenarios.

- **Transformation Process:** The core transformation function, process-bpmn-file, is invoked with the provided input file name. The library meticulously executes the defined transformations, generating a modified BPMN file..

- **Output Information:** The user is informed about the successful completion of the transformation, and the name of the modified BPMN file is presented for reference.

3.0.7 Conclusion:

In the dynamic landscape of BPMN modeling, "bpmn-transform" emerges as a versatile and user-friendly library, dedicated to simplifying the intricate task of dual partner generation. From its basic usage to advanced customization and batch processing capabilities, the library caters to a spectrum of user needs, ensuring flexibility and adaptability. By seamlessly integrating "bpmn-transform" into your BPMN workflow, you unlock a new realm of possibilities, fostering clarity and collaboration in complex business processes.

Chapter 4

Implementation

The implementation of this work consist of the implementation of the theoretical part and the implementation of the library function. This chapter is arranged accordingly.

4.0.1 Theorem:

A sound structured orchestration can be obtained by comparing communication free block, sequential block, global choice block, parallel block and global loop block. Then the collaboration between T and D(T) is sound. Let us denote D(T) the dual of our orchestration flow diagram and T is defined by the transformation

base cases Lemma:

Given a task T_i such that the collaboration between Ti and $D(T_i)$ is sound this also true for every possible way.

Proof: Assume that the flow of T and D(T) have started. Since the collaboration between T_1 and $D(T_1)$is sound we know that their collaboration will be completed and this is also appropriate for every other possible way. Below (Figure 4.1) showing the visualization of base case lemma.

From the diagram illustrate above with a base case such that the collaboration between T_1 and $D(T_1)$ is sound and complete then the collaboration between every other possible combination is also proved as a sound.

Communication free Leema:

Given a sound communication free diagram T then the collaboration between T and D(T) is also sound.

Proof: It is immediate to show that D(T) is the dual of T. T is communication free. Therefore, when T and D(T) start the collaboration they each complete in a sound way. Below (Figure 4.2) showing the visualization of the given lemma.

In a BPMN diagram of communication free such that the collaboration between T_i and $D(T_i)$ are sound then the collaboration between T and D(T) of the figure above is also sound.

Sequential Composition Leema:

Given two process T_1 and T_2 such that both the collaboration between T_1 and $D(T_1)$ and between T_2 and $D(T_2)$ are sound. The collaboration depicted in the figure below which is sound based on the condition carried by the lemma.

Proof: Assume that the flow of T and D(T) have started. Since the collaboration between T_1 and $D(T_1)$ is sound we know that their collaboration will be completed and then the collaboration between T_2 and $D(T_2)$ will start. Since such collaboration is also sound that this is follows. Below (Figure 4.3) showing the visualization of the given lemma.

From the diagram illustrate above with a sequential combination such that the collaboration between T_1 and $D(T_1)$ is sound and complete then the collaboration between T_2 and $D(T_2)$ is also proved as a sound.

Global Choice Leema:

Given two process A_1 and A_2 and two different task T_1 and T_2 such that both the collaboration between T_i and $D(T_i)$ are sound. The collaboration depicted in the figure below which is sound based on the condition carried by the lemma

Proof: Assume T and D(T) are started. Then T need to choose between A_1 and A_2. If T choose A_1 then D(T) will follow the dual of A_1 and since the collaboration between T

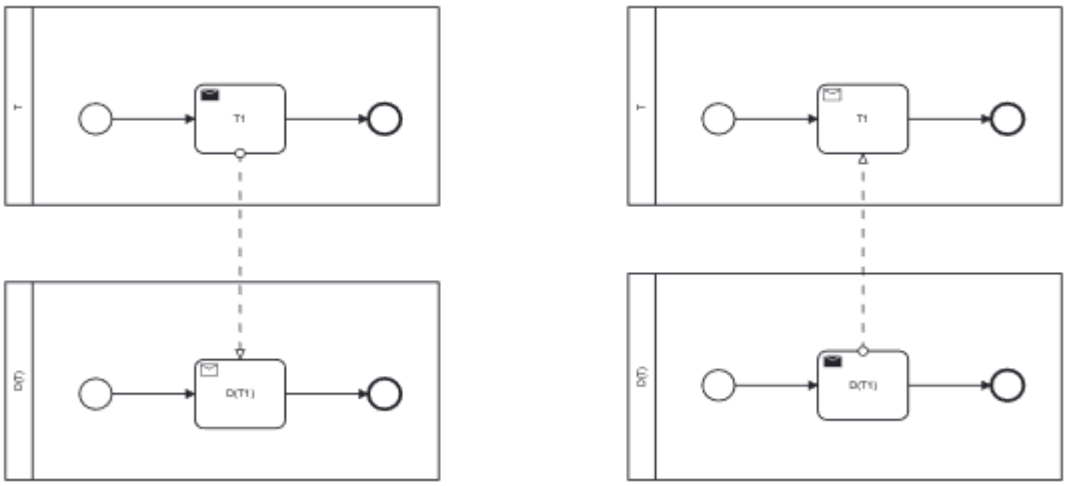

Figure 4.1: Visualization of base case lemma.

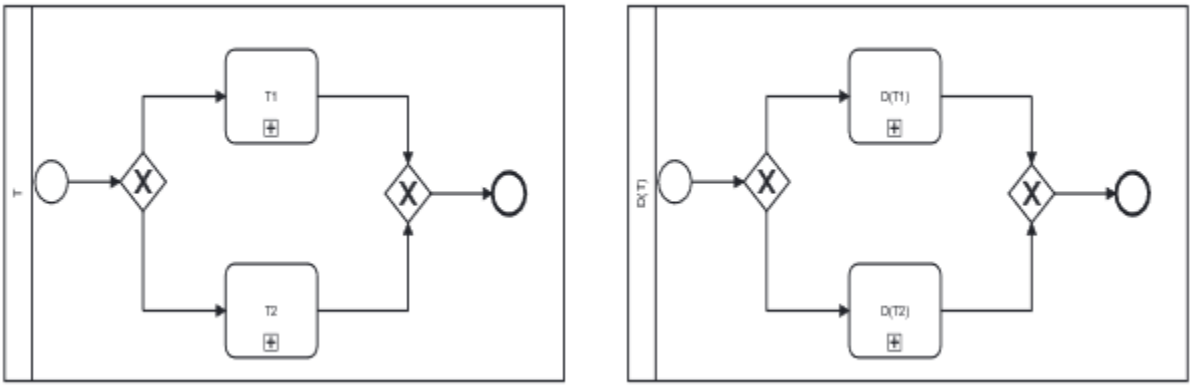

Figure 4.2: Visualization of communication free lemma.

and D(T) is sound the whole collaboration is also sound as well as for A_2. After the collaboration between the task A_1 and $D(A_1)$ further proceedings will happen in the process T_1 and according to the sequential lemma collaboration between T_1 and $D(T_1)$ will also sound. This is also true if the proceedings will happen with T_2. Below (Figure 4.4) showing the visualization of the given lemma.

From the diagram illustrate above with a global choice such that the collaboration between T and D(T) is proved as a sound. Where the collaboration between A_1 and $D(A_1)$ is sound following this the collaboration between A_2 and $D(A_2)$ is also proved as sound. Following this the collaboration between T_1, $D(T_1)$ and T_2, $D(T_2)$ is also proved as sound according to the sequential lemma.

Parallel composition Leema:

Given two process T_1 and T_2 in parallel composition such that both the collaboration between T_1 and $D(T_1)$ and between T_2 and $D(T_2)$ are also sound. The collaboration depicted in the figure below which is sound based on the condition carried by the lemma.

Proof: Assume T and D(T) are started. Then T will start with both T_1 and T_2 and D(T) will start both $D(T_1)$ and $D(T_2)$ since the collaborations between T_i and $D(T_i)$ is sound the whole collaboration is also sound. Below (Figure 4.5) showing the visualization of the given lemma.

From the diagram which is illustrate above with a Parallel composition such that the collaboration between T and D(T) where T start with both the process T_1 and T_2 then the collaboration between T_1 and $D(T_1)$ is sound then the collaboration between T_2 and $D(T_2)$ is sound also proved as a sound.

Global loop Leema:

Given a sound diagram T with a global loop then the collaboration between T and D(T) is also sound.

Proof: It is anticipated that D(T) is the dual of T. Then after achieving T_1, T need to

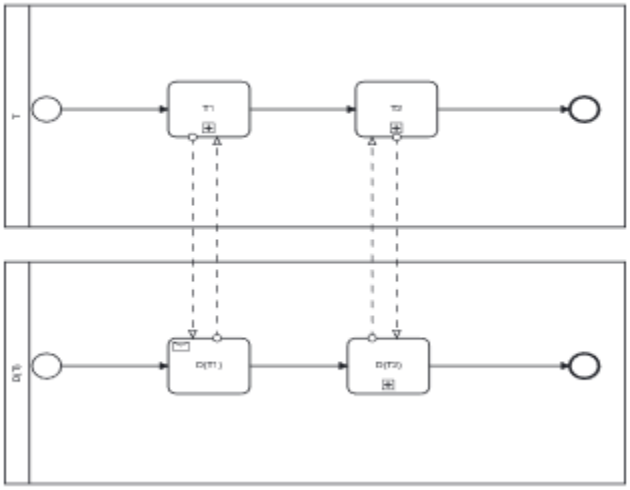

Figure 4.3: Visualization of sequential lemma.

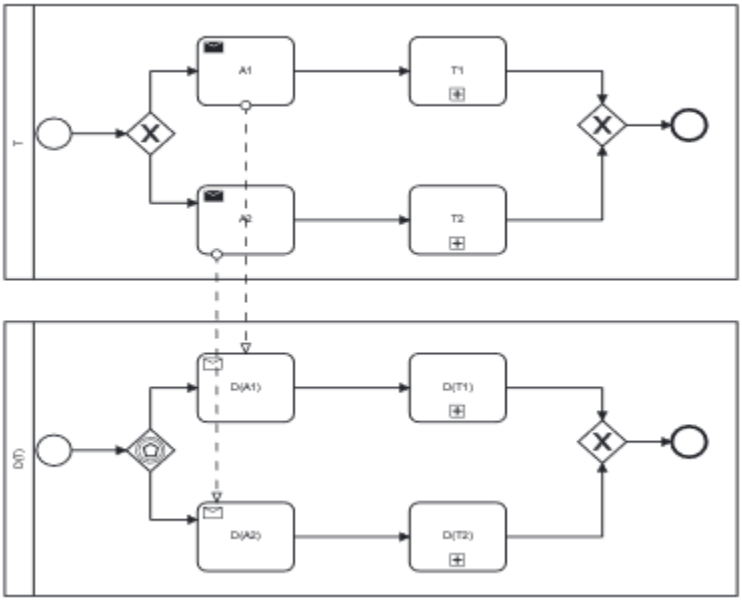

Figure 4.4: Visualization of global choice lemma.

choose between the way exit (E) from the loop or rework (R) again into the loop. where R can join the loop by achieving T2. The collaboration between T_1 and $D(T_1)$ is sound. If T choose E, then D(T) will follow the dual of E which is D(E) and since the collaboration between T and D(T) is sound then the collaboration between E, D(E) is also sound. If the process T choose to rework again with the loop, then the collaboration between R, D(R) and T_2, $D(T_2)$ will sound as per sequential lemma. This proves the whole collaboration between T and D(T) as sound. Below (Figure 4.6) showing the visualization of the given lemma.

From the diagram which is illustrate above with a global loop such that the collaboration between T and D(T) where after achieving the sound collaboration between T_1 and $D(T_1)$ the choice of T_1 exist (E) or rework (R) is happening. So the collaboration between E , D(E) and R, D(R) is also sound While choosing the stay inside the loop the task following R is also sound with its collaboration D(R).

4.0.2 Algorithm for implementing dual partner

This section illustrate the algorithm for implementing the dual partner. As we are generating the dual partner from a given bpmn diagram such that we are modifying the diagram . For this reason we named this diagram **Modify BPMN**

Explanation of the Algorithm

- **Input:** input-file-name: Name of the input BPMN file.
- **Output:** A modified BPMN file with dual partners, saved as input-file-name-modified.bpmn.

1. **Parse the Input BPMN File:** Use the ElementTree module to parse the input BPMN file. Obtain the root element of the XML tree.
2. **Initialize Element Mapping Dictionary:** Create an empty dictionary to track the mapping between old and new BPMN elements.
3. **Task Replacement:**For each element in the BPMN file: If the element is a "sendTask," replace it with a "receiveTask" and vice versa. Update the element mapping dictionary.

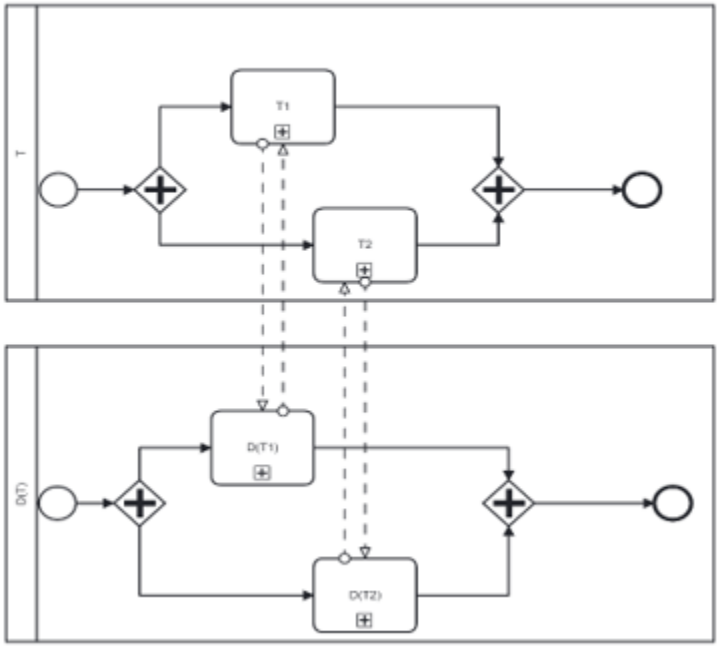

Figure 4.5: Visualization of Parallel composition lemma.

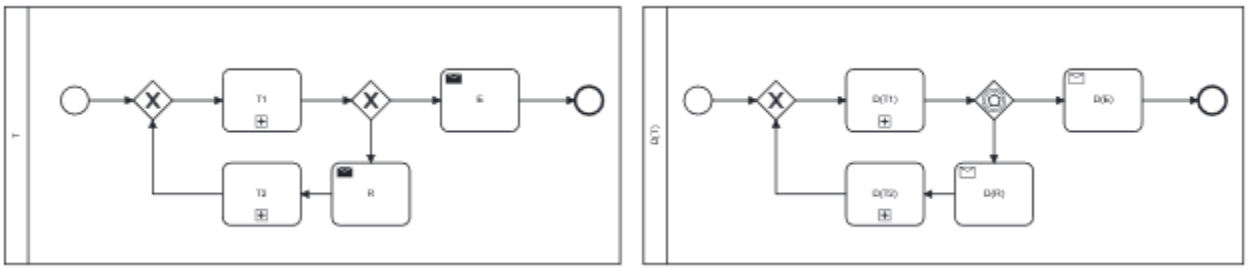

Figure 4.6: Visualization of Global loop lemma.

Algorithm 1 Modify BPMN

```
procedure ModifyBPMN(Tree T, Root R, Mapping M)
    Input: Process BPMN File {Tree (T), Root (R), Mapping (M)}
    Output: Modified BPMN

    procedure ReplaceTask(element)
        for i in element to R do
            if "sendtask" == i then
                n ← i.replace("sendtask", "receivetask")
                M[id] ← i
            else
                n ← i.replace("receivetask", "sendtask")
                M[id] ← i
            end if
        end for
    end procedure

    procedure GatewayReplaces(element)
        for j in element to R do
            if "eventBasedGateway" == j then
                m ← j.replace("eventBasedGateway", "exclusiveGateway")
                M[id] ← j
            else
                m ← j.replace("eventBasedGateway", "exclusiveGateway")
                M[id] ← j
            end if
        end for
    end procedure

    procedure MessageIntermediateEventsReplace(element)
        for k in element to R do
            if "intermediateCatchEvent" == k then
                p ← k.replace("intermediateCatchEvent", "intermediateThrowEvent")
                M[id] ← k
            else
                p ← k.replace("intermediateThrowEvent", "intermediateCatchEvent")
                M[id] ← k
            end if
        end for
    end procedure

    procedure RemoveDataObjects(element)
        for l in element to R do
            if "DataObjectReference" == l and "DataStoreReference" == l then
                o ← l.remove("DataObjectReference" & "DataStoreReference")
                M[id] ← l
            end if
        end for
    end procedure
end procedure
```

4. **Event-Based Gateway Replacement:** For each element in the BPMN file: If the element is an "eventBasedGateway," replace it with an "exclusiveGateway." Update the element mapping dictionary.
5. **Message Intermediate Event Replacement:**For each element in the BPMN file: If the element is an "intermediateCatchEvent," replace it with an "intermediateThrowEvent" and vice versa. Update the element mapping dictionary.
6. **DataObjectReference and DataStoreReference Removal:** For each dataObjectReference element found within the BPMN file, remove it. For each dataStoreReference element found within the BPMN file, remove it.
7. **Sequence Flow Update:For each element in the BPMN file:** If the element is"sequenceFlow," update its sourceRef and targetRef attributes based on the element mapping.
8. **Namespace Replacement:** Manually replace specific namespace prefixes in the root element's tag.
9. **Write the Modified Content to the Output File:**Write the modified XML content to a new file with a modified name.
10. **Complete the Process:**Print a message indicating the successful completion of the transformation, including the name of the modified BPMN file.

End of Algorithm

This algorithm provides a step-by-step guide to the logic implemented in the provided Python code for generating dual partners in a BPMN diagram, without using specific code snippets.

4.0.3 Incorporation with machine learning:

In the realm of business process analysis, the inadequacies of conventional methods have spurred a quest for innovative approaches that transcend the limitations of capturing intricate relationships between activities. The motivation for this research emanates from the

imperative realization that traditional methods hinder the ability to make informed decisions crucial for optimizing processes, allocating resources judiciously, and enhancing overall performance.

Recognizing this gap, our research takes a pioneering step by incorporating the dual partner concept into business process analysis. The dual partner concept introduces a paradigm shift, offering a more holistic and nuanced representation of business processes. This approach promises a deeper understanding of workflow dynamics, aiming to revolutionize the landscape of decision-making in process optimization. The primary objective of this research is to rigorously explore the efficacy of the dual partner concept in elevating business process analysis. Our investigation unfolds with three key focal points:

- **Introduction of Dual Partners:**We embark on the development of a systematic approach to identify and introduce dual partners for each activity within a given business process. This novel concept transcends the conventional, introducing a dynamic element that encapsulates the inherent relationships between activities.

- **Modification of Outcome Predictions:**Leveraging the power of machine learning algorithms, we assess the impact of replacing traditional activity mappings with dual partners on outcome predictions. This facet of the study explores how the incorporation of dual partners influences the accuracy and reliability of predictive modeling within business processes.

- **Validation Across Models:** To comprehensively gauge the adaptability and robustness of the dual partner concept, our research extends beyond machine learning to include simulation, optimization, and reinforcement learning models. This multi-model validation approach ensures a thorough examination of the dual partner concept's efficacy in diverse analytical frameworks.

A key distinguishing factor of our study lies in its reliance on a diverse dataset derived from real-world business processes spanning industries such as finance, e-commerce, and manufacturing. This intentional selection allows for a comprehensive evaluation of the dual

partner concept's applicability across different domains, ensuring the generalizability and practicality of our findings. As we delve into the subsequent sections, the intricacies of our research methodology and the novel insights derived from the integration of the dual partner concept will unfold. By addressing the inherent limitations of traditional business process analysis, this research endeavors to contribute significantly to the evolution of decision-making strategies, setting the stage for a more informed, adaptive, and effective approach to business process optimization.

Augmenting BPMN Processes with Dual Partners

In the ever-evolving landscape of Business Process Model and Notation (BPMN), the quest for efficiency and adaptability has led us to explore groundbreaking intersections between BPMN processes and machine learning. This journey is marked by the integration of a novel concept.The generation of dual partners for a given BPMN process. The driving force behind this innovative approach is the belief that enriching BPMN processes with dual partners can unlock new dimensions of flexibility, resilience, and optimization.

In the realm of business process management, the challenge has always been to enhance processes dynamically, adapting them to the ever-changing demands of the contemporary business environment. Traditional methods often fall short in addressing the need for adaptability and resilience in BPMN processes, especially when confronted with unforeseen variations or disruptions. It is within this context that our pursuit of a transformative solution takes shape – the creation of dual partners for BPMN processes, coupled with the power of machine learning.The core idea behind generating dual partners for BPMN processes stems from the recognition that a singular, rigid workflow may not always suffice in a dynamic business landscape. A dual partner, in this context, is a counterpart to a specific activity in the BPMN process, providing an alternative route or decision point. This innovative approach opens avenues for increased agility, improved decision-making, and enhanced process resilience.

Consider a scenario where an order placement activity encounters an unexpected delay. In a traditional BPMN process, such a delay could have a cascading impact on subsequent activities. However, by introducing a dual partner – an alternative order fulfillment path, for instance – the process gains the ability to seamlessly adapt to unforeseen circumstances, ensuring a smoother and more resilient workflow.

The integration of machine learning into this paradigm marks a pivotal advancement. By harnessing the capabilities of machine learning models, we empower our BPMN processes with the intelligence to dynamically engage dual partners based on real-time data, historical patterns, and predictive analytics. This fusion of process dynamics and machine learning prowess becomes a catalyst for a paradigm shift in how we conceptualize, implement, and optimize business processes.

In the code provided, we witness the practical implementation of this synergy. Augmenting the BPMN process involves not only the generation of dual partners but also the strategic utilization of machine learning to simulate, optimize, and even predict the outcomes of these augmented processes. The code elegantly balances the intricacies of time perturbation, duration variations, and feature engineering, laying the foundation for a robust and adaptable BPMN framework.

The question naturally arises – why embark on this unique integration of dual partners and machine learning in BPMN processes? The answer lies in the manifold benefits that this approach bestows upon organizations:

- **Enhanced Flexibility and Adaptability:** By introducing dual partners, BPMN processes become inherently flexible, capable of adapting in real-time to external disruptions, market variations, or unforeseen delays. This agility is paramount in the contemporary business landscape where change is the only constant.

- **Resilience in the Face of Uncertainty:** The ability to seamlessly switch to dual partners in response to unexpected events ensures that the BPMN process remains

resilient. Unforeseen challenges do not translate into bottlenecks, as the system intelligently navigates through alternative pathways.

- **Optimized Decision-Making:**Machine learning, integrated into the augmented processes, contributes to optimized decision-making. Predictive analytics and historical data guide the system in choosing the most effective dual partner, enhancing overall efficiency.

- **Continuous Improvement Through Simulation:** The code's simulation model, powered by machine learning, facilitates continuous improvement. By mimicking real-world scenarios and predicting outcomes, organizations can fine-tune their processes, making them more efficient and adaptive over time.

- **Empowering Human-Centric BPMN:** This integration is not about replacing human decision-makers but empowering them. Dual partners, guided by machine learning insights, provide decision-makers with valuable alternatives, enabling more informed and strategic choices.

However, the marriage of dual partners and machine learning in BPMN processes heralds a new era of adaptability and efficiency. As we delve into the intricacies of the code provided, we witness the tangible manifestation of this vision. This isn't merely a technical innovation; it's a paradigm shift that empowers organizations to navigate the complexities of the modern business landscape with unprecedented agility and foresight.

4.0.4 Methodology of Integrating Dual Partners and Machine Learning for Enhanced BPMN Processes

The methodology for incorporating dual partners into BPMN processes, augmented by machine learning, follows a systematic and comprehensive approach. This innovative fusion is designed to enhance adaptability, resilience, and decision-making within business processes. The presented methodology encompasses data augmentation, cleaning, feature engineering, exploratory data analysis (EDA), and the integration of machine learning models, all of which are implemented through the provided code.

- **Data Augmentation:** The first step involves augmenting the original BPMN process data. This is achieved through the augment-data function, which generates additional samples by perturbing the timestamp and duration of each entry in the dataset. The objective is to introduce variability and simulate real-world scenarios, providing the foundation for a more robust and adaptive BPMN framework. The augmentation process considers time and duration variations, injecting a level of unpredictability into the dataset. The number of augmented samples, as well as the standard deviations for time and duration perturbations, can be customized to suit the specific requirements of the BPMN process under consideration.

- **Cleaning and Feature Engineering:** The augmented dataset undergoes a thorough cleaning and feature engineering process using the clean and engineer features function. This phase ensures data quality, addresses missing values, and enhances the dataset with additional features that contribute to the overall adaptability and intelligence of the BPMN processes. Key steps include handling missing values by dropping rows with null entries, converting the timestamp to datetime format, and ensuring the 'duration' column is of numeric type. Feature engineering steps involve extracting date components from the timestamp, creating binary features, encoding categorical variables, and scaling numerical features using the Standard Scaler. Below (Figure 4.7) showing the cleaning and feature engineering process.

  The flowchart above illustrates the sequence of steps involved in cleaning and feature engineering. Each step, from handling missing values to scaling numerical features, is visually depicted for clarity.

- **Exploratory Data Analysis (EDA):** The exploratory data analysis component involves visualizing the dataset to gain insights into the distribution of categorical attributes. The code includes a 3D histogram plot using matplotlib, showcasing the frequency of different attribute values. This EDA step provides a holistic view of the dataset's composition, aiding in understanding the patterns and relationships between

various attributes. Below (Figure 4.8) showing the Exploratory data analysis process.

A 3D histogram plot using matplotlib, presenting the distribution of categorical attributes. This visualization aids in understanding attribute frequency and patterns within the dataset.

- **Machine Learning Modeling:** The methodology includes the implementation and evaluation of machine learning models for both simulation and classification tasks. The code provides a framework for employing Random Forest models, a versatile choice known for its adaptability to various scenarios.For simulation, a Random Forest Regressor is trained to predict outcomes based on the augmented and cleaned dataset. This simulation model captures the dynamic behavior of the BPMN processes and serves as a basis for optimization and decision-making.For classification, a Random Forest Classifier is employed, leveraging the power of machine learning to make informed decisions based on historical data and feature engineering. GridSearchCV is utilized for hyperparameter tuning, ensuring optimal model performance. Below (Figure 4.9) showing the analysis of the machine learning modelling process.

An illustration depicting the Random Forest Classifier used for classification. The model leverages machine learning to make informed decisions, optimizing performance through hyper parameter tuning

- **Implementation of Dual Partner and Evaluation**5. : The methodology introduces the concept of dual partner replacements within the BPMN process. A mapping is created to replace certain activities with their dual counterparts, enabling the system to dynamically adapt to disruptions. To assess the impact of dual partner replacements, we trained and tested the optimized Random Forest Classifier on the modified dataset.

Classification Report on Validation Set and Testing Set: below (Figure 4.10) present the classification reports for the validation set and testing set, respectively.

These reports provide detailed metrics such as precision, recall, F1-score, and support, offering a comprehensive view of the model's performance on both datasets.

The classification reports reveal that the model achieves a remarkable accuracy which is 1. This is a notable result; however, it is important to emphasize that achieving 1 accuracy is expected in this context. The model's ability to predict dual partners is crucial for the successful implementation of the proposed approach. Without good accuracy, the model might misinterpret or fail to recognize the dual partners, impacting its adaptability to disruptions within the BPMN process.

Learning Curve based on Accuracy and Training Set Size: below (Figure 4.11) depicts the learning curve, showcasing how the accuracy of the Random Forest Classifier evolves with varying training set sizes. This curve provides insights into the model's ability to generalize as the amount of training data increases.

Epochs and Loss of Training Loss and Validation Loss: (Figure 4.12) represent the visualization of the training and validation loss across epochs during the model's training phase. Monitoring loss over epochs helps in understanding the convergence behavior and potential over fitting or under fitting. From the figure we can see that there is not any visualization of over fitting or under fitting.

The combination of these diagrams and model evaluation metrics, including accuracy and confusion matrix, provides comprehensive insights into the effectiveness of dual partners in enhancing decision-making and adaptability within the BPMN process.

Logging and Documentation: The methodology underscores the utmost importance of meticulous logging to capture changes made during the dual mapping process within the BPMN framework. A dedicated log file is systematically generated, providing a detailed account of critical information, including timestamps, original activities, dual activities, and the nature of the action (dual mapping). This log file stands as

a valuable repository, facilitating the tracking of modifications and offering profound insights into the evolution of the BPMN process. The log file entries exemplify specific instances of dual mapping, where activities undergo transformation as indicated in the sample entries through the below (Figure 4.13)

For instance, consider the entries related to 'activity-Order Placement' and 'activity-Order Fulfillment.' In the BPMN context, these activities represent a 'send task' and 'receive task,' respectively. The log file explicitly illustrates the logical application of dual mapping, wherein the 'send task' is logically replaced by the 'receive task.' This aligns seamlessly with the earlier conceptualization that a 'send task' becomes its dual counterpart, a 'receive task,' enhancing the adaptability and resilience of the BPMN process.This structured documentation not only serves as a record but also provides clarity on the logical transformations applied during dual mapping, contributing significantly to the understanding of the BPMN process's evolutionary journey

Continuous Improvement Through Simulation: The simulation model, powered by machine learning, becomes a tool for continuous improvement. By mimicking real-world scenarios and predicting outcomes, organizations can iteratively fine-tune their processes, ensuring that the BPMN framework evolves to meet changing business dynamics. This comprehensive methodology, exemplified by the provided code, represents a pioneering approach to synergizing BPMN processes with machine learning intelligence. By seamlessly integrating dual partners and predictive analytics, organizations can navigate the complexities of the business landscape with unprecedented adaptability and foresight.

the integration of dual partners into BPMN processes, complemented by the power of machine learning, represents a groundbreaking leap towards a more adaptive, resilient, and intelligent business process management framework. The methodology, as demonstrated through the provided code, has showcased the potential to enhance decision-making, mitigate the impact of disruptions, and continuously refine processes

based on real-world simulations.The augmentation of BPMN processes through dual partners introduces a level of flexibility that is paramount in the dynamic landscape of contemporary business. The synergy with machine learning not only enables predictive decision-making but also fosters a culture of continuous improvement, where processes evolve in response to changing circumstances.As we reflect on this transformative journey, the significance of dual partners becomes evident in their ability to act as dynamic counterparts, steering the BPMN processes through unforeseen challenges. Machine learning, seamlessly integrated into this paradigm, emerges as a guiding force, providing valuable insights and predictive capabilities that empower organizations to navigate complexities with foresight and adaptability.

While the current methodology lays a solid foundation, there are avenues for future exploration and enhancement. Exploration of algorithms and strategies for the dynamic allocation of dual partners based on real-time data. This could involve machine learning models that adapt and learn from ongoing process performance. Also Investigate the application of more advanced machine learning models, such as deep learning architectures, to capture intricate patterns and dependencies within BPMN processes. Development capabilities for real-time simulation, allowing organizations to respond to changes promptly. This involves refining the simulation model to provide instantaneous insights into potential outcomes. One of the most important future works on further research into incorporating human-centric adaptability by considering user feedback and preferences in the decision-making process. This could involve reinforcement learning techniques. The scalability and generalization of the methodology across different industries and business domains. Consideration should be given to diverse BPMN processes and their unique requirements. Explore integration with external systems and data sources to enrich the decision-making process. This could involve leveraging external data for a more comprehensive understanding of contextual factors. Develop a user interface that provides decision-makers with an intuitive platform to interact with dual partners and machine learning insights. This enhances the user experience and promotes user adoption.

4.0.5 Generation of the library function of the dual partner:

In the dynamic realm of Business Process Model and Notation (BPMN), the innovative idea of Dual Partner Generation is poised to revolutionize the way we conceive, design, and adapt business processes. The genesis of this transformative approach lies in a meticulously crafted library function, introducing the novel capability of generating dual partners within BPMN processes – an uncharted territory that holds the promise of unparalleled adaptability. The crux of Dual Partner Generation lies in its ability to dynamically replace and interchange BPMN elements such as tasks, gateways, and events. The underlying logic orchestrates a systematic transformation that goes beyond traditional BPMN modeling constraints. Through a series of well-defined steps, the library function brings forth a paradigm shift, enabling the automatic generation of dual partners. We can make a breakdown of the ideas by following.

- Meticulously identifying and replacing send tasks with their dual counterparts, the receive tasks, and vice versa, introduces a layer of adaptability, allowing tasks to seamlessly shift their roles in response to evolving process dynamics.

- Beyond tasks, the library function extends its transformative touch to gateways, specifically event-based gateways. By intelligently replacing these gateways with exclusive gateways, it ensures that the BPMN process gains a heightened level of decision-making flexibility.

- Further augmenting the adaptive prowess of the BPMN process, the library function facilitates the interchangeability of intermediate events. It dynamically switches between catch and throw events, fostering a dynamic environment where events respond to triggers with unparalleled fluidity.

- To ensure the seamless integration of modified BPMN files, the library function undertakes a critical step in adjusting namespaces. This ensures not only the technical integrity of the BPMN model but also its compatibility within established BPMN ecosystems.

The introduction of dual partners within BPMN processes marks a strategic departure from static, predefined structures. This innovation empowers BPMN models to navigate disruptions and changes with unprecedented agility. The inherent adaptability afforded by dual partner generation becomes a strategic asset, allowing businesses to respond proactively to market shifts, regulatory updates, or internal restructuring. As we delve into the intricacies of the library function-driven Dual Partner Generation, we embark on a journey that transcends the conventional boundaries of BPMN. This introduction serves as a portal to a realm where BPMN processes not only model business activities but actively engage with the dynamic landscape they inhabit. Dual Partner Generation is not just a code – it is a key unlocking the door to a new era of responsive, resilient, and future-ready BPMN processes.

Methodology The methodology for implementing Dual Partner Generation is meticulously crafted, guided by the transformative ideas of dual partners. This systematic framework reshapes the conventional BPMN landscape, offering a structured approach for generating and adapting dual partners within business processes.

- **Importing Essential Libraries:** The process commences with the importation of crucial libraries, notably the ElementTree module for XML parsing. This foundational step ensures that the subsequent implementation aligns seamlessly with the BPMN file structure, providing a robust foundation for the methodology.

- **Creating the process-bpmn-file Function:** At the heart of the methodology lies the creation of the process-bpmn-file function. This function encapsulates the entire process, leveraging the ideas of dual partners to introduce dynamic changes to BPMN elements. By importing the XML file of a BPMN diagram, the function establishes the groundwork for transformative modifications.

- **Procedural Logic for Dual Partner Generation:** The function systematically replaces tasks, adapts gateways, and facilitates the interchangeability of intermediate events – all in accordance with the principles derived from the ideas of dual partners. Each step is meticulously designed to enhance the adaptability and responsiveness of the BPMN process.

- **Local Generation of the Library Function:** The implementation of Dual Partner Generation is encapsulated within a library function, a localized generation tailored to the specific needs of the user. This ensures that the ideas of dual partners are not only conceptual but are translated into a tangible, user-friendly tool that can be seamlessly integrated into diverse BPMN environments.

- **Utilizing the Localized Library Function:** Once generated locally, the library function becomes a versatile tool that can be easily utilized. By incorporating this function into existing BPMN workflows, users gain the ability to dynamically generate dual partners, ushering in a new era of adaptability and agility within their business processes.

- **Output:** Modified BPMN Diagram: The culmination of the methodology is the generation of the modified BPMN diagram. The process-bpmn-file function outputs a transformed representation of the original BPMN process, infused with the ideas of dual partners. This modified diagram encapsulates the adaptability and dynamism introduced by the dual partner generation process.

- **Integration and Local Deployment:** The methodology acknowledges the practicality of integrating the library function seamlessly into existing workflows. By locally deploying this user-friendly tool, organizations can harness the power of dual partner generation, aligning their BPMN processes with the dynamic demands of the contemporary business landscape.

4.0.6 Conclusion

As we traverse through the steps of this methodology, it becomes evident that the implementation of dual partners within BPMN processes is not just a theoretical construct. It is a tangible, actionable process driven by a library function that embodies the transformative ideas of dual partners. This methodology is not merely a set of instructions; it is a roadmap guiding organizations toward a future-ready BPMN paradigm, where adaptability is not just a concept but a realized capability.

From the (Figure 4.14) above we need to make the dual partner for it. while generating the dual partner it contains a suffix "-modified". Below the diagram represent the dual partner for our generated bpmn diagram. The (Figure 4.15) illustrates the transformation from a conventional BPMN diagram to its dual partner counterpart. The dynamic changes introduced through the dual partner generation process are visualized, showcasing the adaptability infused into the business process. For importing the library below code in python which is illustrating the (Figure 4.16) is required This (Figure 4.16) provides an overview of the library function is utilized to create the dual partner. The seamless integration of the library into the code highlights the simplicity and effectiveness of incorporating dual partners within the BPMN framework.

4.0.7 Conclusion

In conclusion, the journey through Dual Partner Generation unfolds as a paradigm shift in Business Process Model and Notation (BPMN). The integration of dual partners into BPMN processes is not merely a technological advancement; it is a strategic imperative for organizations seeking unparalleled agility and adaptability in their operational frameworks. The implementation of the process-bpmn-file function and the local generation of the library function signify more than just a technological innovation – they embody a transformative approach towards BPMN design. By infusing the ideas of dual partners into the heart of business processes, organizations can break free from the constraints of static workflows and embrace a dynamic, responsive future. As organizations embrace Dual Partner Generation, they embark on a journey towards a future where business processes are not rigid structures but living entities capable of adapting to dynamic circumstances. The vision of dual partner generation is realized through the tangible outputs of the methodology – BPMN diagrams that seamlessly incorporate adaptability and responsiveness. The benefits derived from Dual Partner Generation extend beyond the realm of theoretical advantages. Organizations leveraging this methodology unlock the potential for rapid response to disruptions, enhanced decision-making capabilities, and a holistic transformation of their BPMN ecosystems. The ideas of dual partners become a catalyst for ushering in a new era of operational excellence. While the current implementation of Dual Partner Generation marks a significant leap for-

ward, the journey does not end here. Future endeavors aim to amplify the impact of dual partners by exploring additional dimensions of adaptability and intelligence within BPMN processes. The roadmap for future work encompasses the following key areas: Future iterations will delve into refining and expanding dual partner strategies. The goal is to introduce more nuanced approaches for dual partner generation, allowing organizations to tailor their adaptability based on specific business contexts. The convergence of Dual Partner Generation with cutting-edge technologies, such as artificial intelligence and machine learning, presents an exciting avenue for exploration. Integrating intelligent decision-making capabilities into the dual partner framework could elevate BPMN processes to unprecedented levels of sophistication. Recognizing the significance of user adoption, future work will focus on developing intuitive, user-friendly interfaces for the library function. The aim is to democratize dual partner generation, making it accessible to a broader audience without specialized technical expertise.

To facilitate widespread adoption, future efforts will prioritize comprehensive documentation and training resources. Empowering users with the knowledge and understanding of dual partner concepts will be instrumental in maximizing the benefits of this innovative approach. Envisioning a collaborative future, the exploration extends to the establishment of a dual partner ecosystem. This ecosystem would foster collaboration, knowledge-sharing, and the development of best practices among organizations embracing dual partner generation. As the journey of Dual Partner Generation unfolds, the landscape of BPMN stands on the brink of a transformative era. The methodologies, ideas, and innovations encapsulated in this endeavor not only redefine the possibilities of business process modeling but also pave the way for a future where adaptability is not a feature – it is the essence of operational resilience. The road ahead is illuminated by the beacon of dual partners, guiding organizations towards a future where business processes evolve in tandem with the dynamic pulse of the digital age.

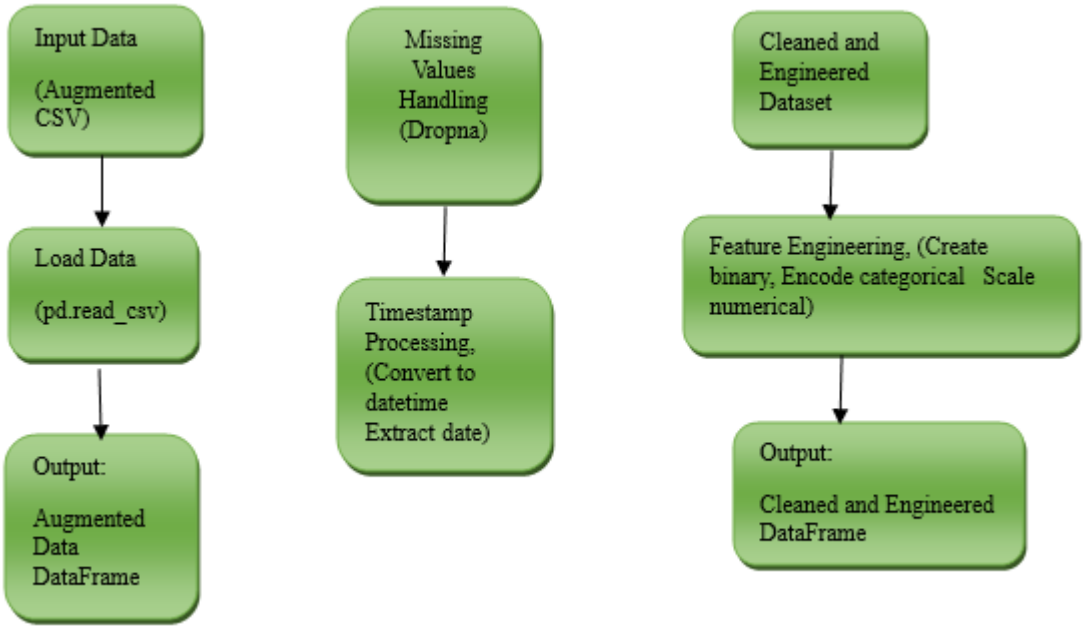

Figure 4.7: Cleaning and Feature Engineering Pipeline).

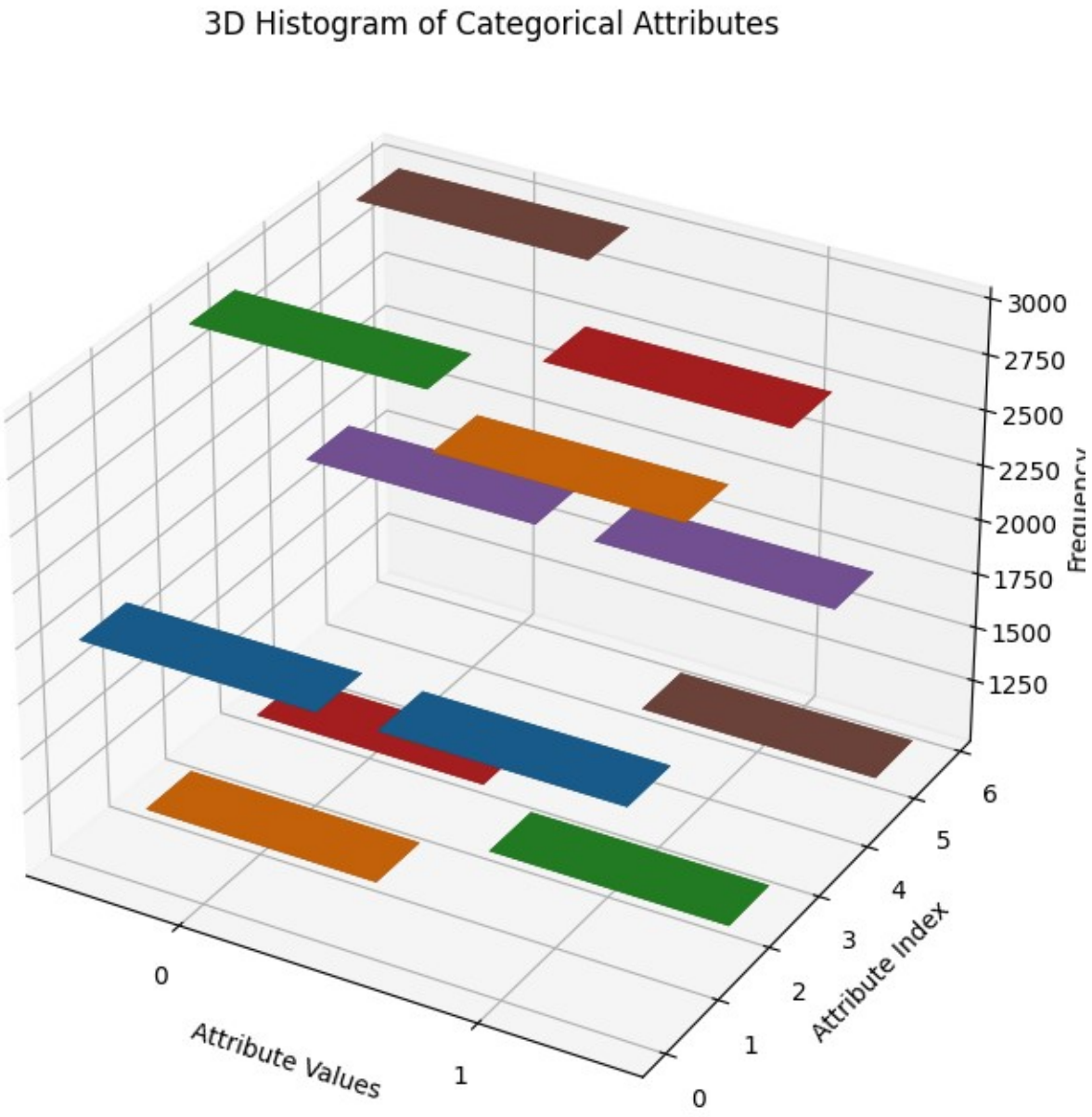

Figure 4.8: Exploratory Data Analysis).

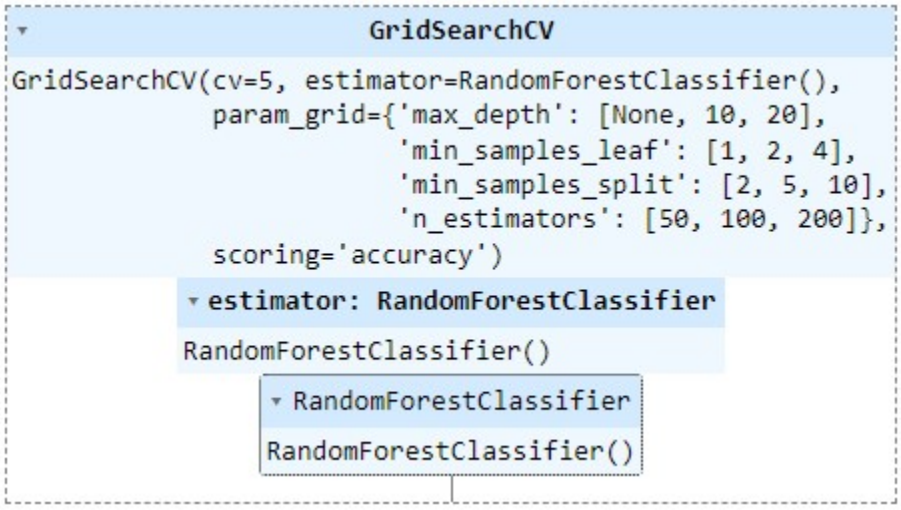

Figure 4.9: Machine learning modelling).

Classification Report on Validation Set:

	precision	recall	f1-score	support
In Progress	1.00	1.00	1.00	201
Success	1.00	1.00	1.00	601
accuracy			1.00	802
macro avg	1.00	1.00	1.00	802
weighted avg	1.00	1.00	1.00	802

Classification Report on Test Set:

	precision	recall	f1-score	support
In Progress	1.00	1.00	1.00	194
Success	1.00	1.00	1.00	608
accuracy			1.00	802
macro avg	1.00	1.00	1.00	802
weighted avg	1.00	1.00	1.00	802

Figure 4.10: Classification Report on Validation Set and Testing Set).

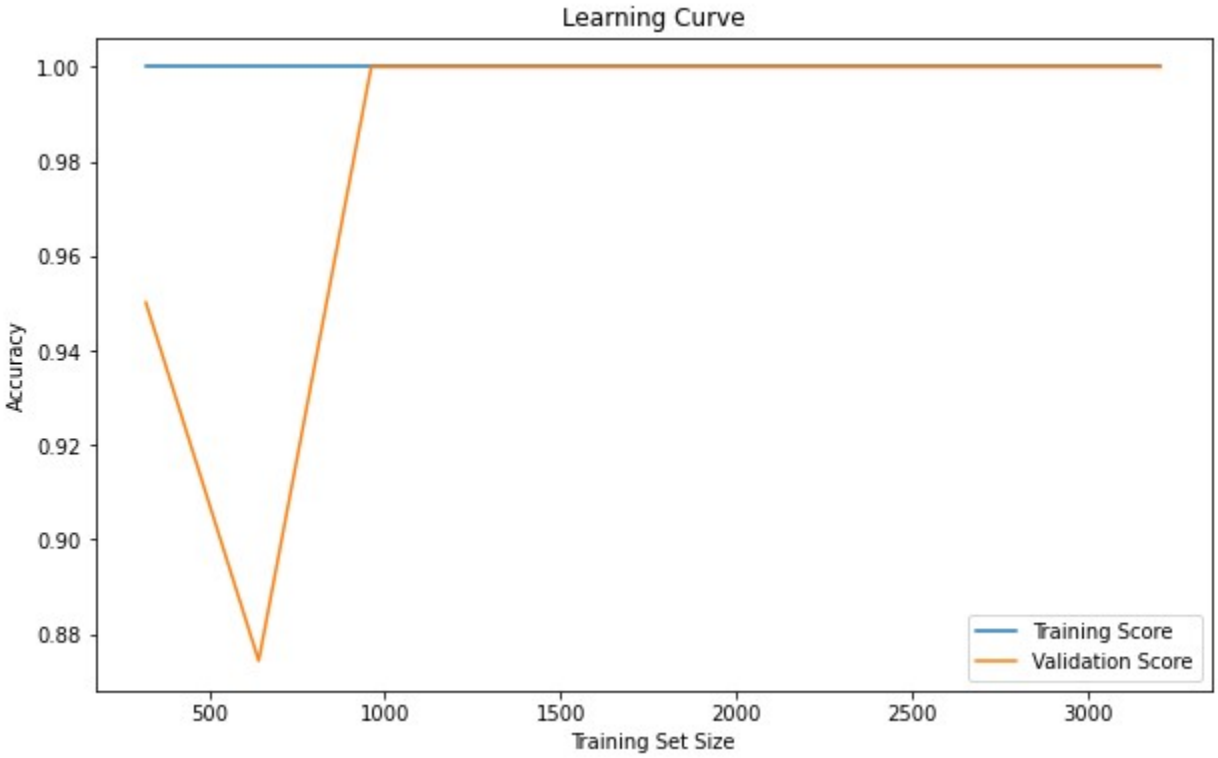

Figure 4.11: Learning Curve based on Accuracy and Training Set Size).

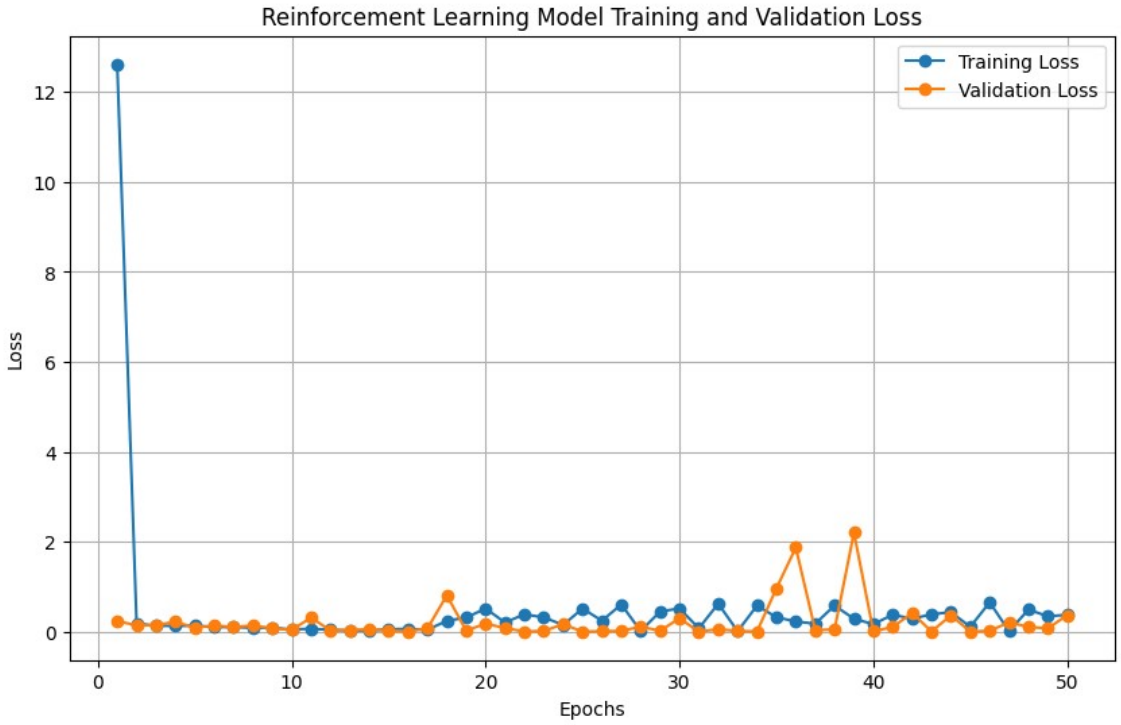

Figure 4.12: Epochs and Loss of Training Loss and Validation Loss

	timestamp	original_activity	dual_activity	action
2	58:23.2	activity_Order Placement	activity_Order Fulfillment	dual_mapping
3	58:23.2	activity_Payment Processing	decision_Approved	dual_mapping
4	58:23.2	decision_Completed	decision_Processing	dual_mapping

Figure 4.13: Snippet of Generated log file

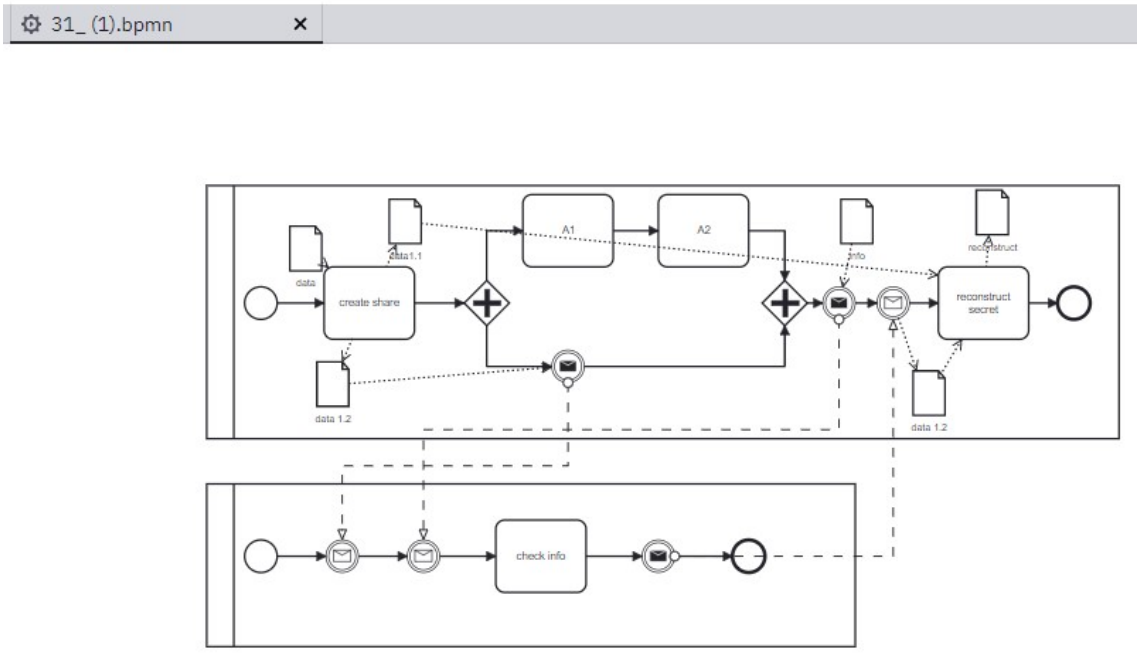

Figure 4.14: Given BPMN diagram

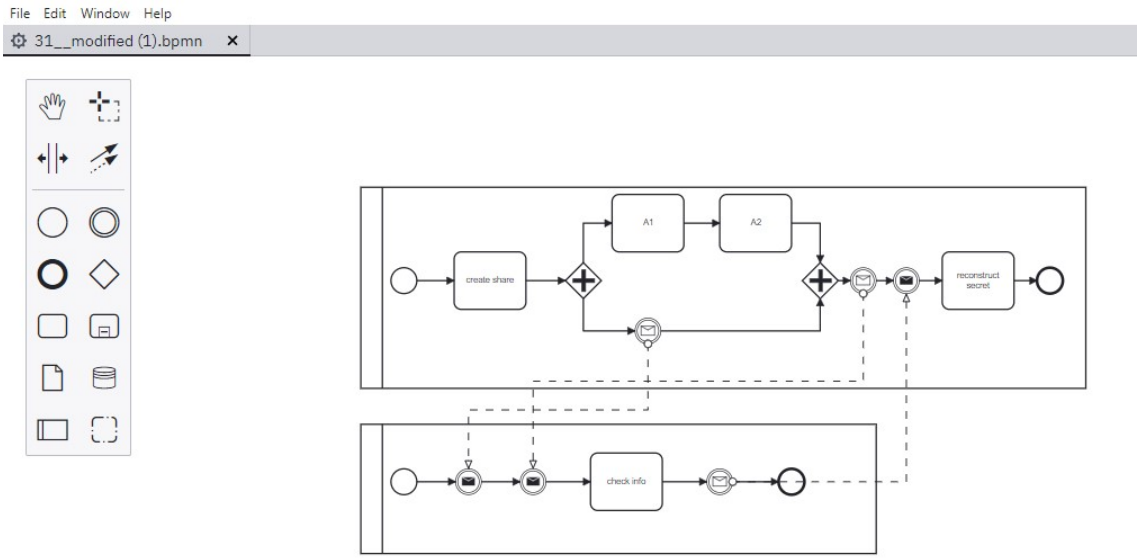

Figure 4.15: Dual partner from the given BPMN diagram

```
In [2]: #from my_bpmn_package import bpmn_transfrom
        from my_bpmn_package.bpmn_transfrom import process_bpmn_file
```

Figure 4.16: code for importing library

Chapter 5

Testing

5.0.1 Introduction:

In the relentless pursuit of refining and validating the ideas of Dual Partner Generation, the testing phase emerges as a critical juncture. Testing not only validates the functionality of the library function but also explores its adaptability across a diverse array of real-world BPMN diagrams. "RePROSitory", a treasure trove of BPMN processes, becomes the testing ground for evaluating the robustness and scalability of Dual Partner Generation. This chapter delves into the intricacies of testing, unveiling the strategies employed to ensure the efficacy of the library function.

Crafting a Rigorous Testing Framework The methodology employed for testing Dual Partner Generation is a meticulous blend of precision and comprehensiveness. The process begins with the selection and download of BPMN diagrams from "RePROSitory", representing a spectrum of industries and complexities. The library function, strategically positioned in a directory accessible via the command prompt, becomes the linchpin for testing.

- **Selection of Test Cases:** The diversity of BPMN diagrams on "RePROSitory" serves as a rich tapestry for test case selection. Carefully curated diagrams are chosen to encompass various BPMN elements, including tasks, gateways, events, and data objects.

- **Batch Processing Script:** A Python script, residing in the desktop directory, facilitates the batch processing of multiple BPMN diagrams. This script accepts compressed files (.rar and .zip) containing BPMN diagrams as input, processes each diagram using

the Dual Partner Generation library function, and outputs a compressed file containing the modified diagrams.

- **Command-Line Execution:** Test execution is orchestrated through the command prompt, enabling seamless interaction with the testing script. Users input a command specifying the location of the input file (containing BPMN diagrams) and the desired output location for the modified diagrams. Below diagram show the command line required for the execution

python process-zip-bpmn.py input-zipfile.zip output-zipfile.zip

5.0.2 Conclusion

The conclusion drawn from the testing phase transcends mere validation; it is a revelation of insights that fortify the reliability and versatility of Dual Partner Generation. As the command prompt orchestrates the execution of the testing script, the modified BPMN diagrams stand as a testament to the adaptability and scalability of the library function. The meticulous selection of test cases from "RePROSitory" mirrors the real-world scenarios that organizations encounter in their BPMN processes. Dual Partner Generation, when subjected to this battery of tests, not only emerges unscathed but showcases its prowess in dynamically adapting to diverse BPMN landscapes.

Chapter 6

Future work:

Advanced Usage Scenarios: While the basic usage provides a quick introduction, "bpmn-transform" accommodates advanced scenarios, enabling users to tailor dual partner generation to specific needs.

- **Customizing Element Replacement Rules:** The library recognizes that different BPMN scenarios may demand unique element replacement strategies. Through advanced configurations, users can define their rules for replacing tasks, gateways, and events, offering unparalleled flexibility.

```
from bpmn_transform import process_bpmn_file,
CustomElementReplacementConfig
custom_config = CustomElementReplacementConfig(
task_replacement_rules={"sendTask":
"customReceiveTask", "receiveTask": "customSendTask"},
gateway_replacement_rules={"eventBasedGateway":"customExclusiveGateway"})
if __name__ == "__main__":
input_file_name = input("Enter the name of the input BPMN file: ")
process_bpmn_file(input_file_name,
custom_config)
print(f"Modified BPMN file saved as{input_file_name.replace('.bpmn',
'_modified.bpmn')}")
```

In this example, the user introduces a CustomElementReplacementConfig instance, specifying custom rules for replacing tasks and gateways. This empowers users to adapt the library to the unique needs of their BPMN processes.

- **Batch Processing and Automation:** For scenarios where bulk transformation is required or automation is preferred, "bpmn-transform" supports batch processing. Users can leverage the library in scripts, workflows, or automation pipelines to ensure consistent dual partner modeling across multiple BPMN files.

```
from bpmn_transform import process_bpmn_file, CustomReplacement
import os
def batch_process_bpmn_files(input_folder):
bpmn_files = [f for f in os.listdir(input_folder)
if f.endswith(".bpmn")]
for file in bpmn_files:
file_path = os.path.join(input_folder, file)
process_bpmn_file(file_path, custom_config)
print(f"Modified file saved as {path.replace('.bpmn','_m.bpmn')}")
if __name__ == "__main__":
input_folder = input("Enter the path to the folder
containing BPMN files: ")
batch_process_bpmn_files(input_folder)
```

In this scenario, the user defines a batch-process-bpmn-files function that processes all BPMN files within a specified folder. This illustrates the adaptability of "bpmn-transform" for diverse usage contexts.

Appendix A

Appendix

A.0.1 Instruction for installing the library function locally:

1. Keep this "my-bpmn-library" folder in a feasible place where all the code can be easily manageable

2. Install setuptools and wheel (if not already installed):command: "pip install setuptools wheel"

3. **Create a Source Distribution:** Open a command prompt or terminal, navigate to the root directory of the project (my-bpmn-library), where the setup.py file is located. For example, if you have your project in the "/path/to/development/my-bpmn-library/ "directory, use the "cd" command to navigate to that directory: and run the following command to create a source distribution with the command:

   command : "python setup.py sdist"

   This will generate a dist directory containing a source distribution of your package.

4. **Create a Wheel Distribution:** To create a wheel distribution (a binary distribution that can be installed faster), run the following command **command: "python setup.py bdist-wheel"**

5. **Distribute Package:** install the package locally for testing by running the following command in the project's root directory **command: "pip install"**

6. open a python script to the same directory where the library is then the code should be use for parsing the library

```python
import xml.etree.ElementTree as ET

def process_bpmn_file(input_file_name):
    tree = ET.parse(input_file_name)
    root = tree.getroot()

    # Create a dictionary to track the old-to-new element mapping
    element_mapping = {}

    # Step 1: Replace tasks
    for element in root.iter():
        if "sendTask" in element.tag:
            new_tag = element.tag.replace("sendTask", "receiveTask")
            element.tag = new_tag
            element_mapping[element.get("id")] = element
        elif "receiveTask" in element.tag:
            new_tag = element.tag.replace("receiveTask", "sendTask")
            element.tag = new_tag
            element_mapping[element.get("id")] = element

    # Step 2: Replace event-based gateways after task replacement
    for element in root.iter():
        if "eventBasedGateway" in element.tag:
            new_tag = element.tag.replace("eventBasedGateway", "exclusiveGateway")
            element.tag = new_tag
            element_mapping[element.get("id")] = element

    # Step 3: Replace message intermediate events
    for element in root.iter():
        if "intermediateCatchEvent" in element.tag:
            new_tag = element.tag.replace("intermediateCatchEvent", "intermediateThrowEvent")
            element.tag = new_tag
            element_mapping[element.get("id")] = element
        elif "intermediateThrowEvent" in element.tag:
            new_tag = element.tag.replace("intermediateThrowEvent", "intermediateCatchEvent")
            element.tag = new_tag
            element_mapping[element.get("id")] = element
```

Figure A.1: Snippet of Code for generating Dual Partner

```python
# Step 4: Remove DataObjectReference and DataStoreReference
for data_object_ref in root.findall('.//{http://www.omg.org/spec/BPMN/20100524/MODEL}dataObjectReference'):
    parent_element = tree.find('.//{http://www.omg.org/spec/BPMN/20100524/MODEL}process')  # Assuming data objects are within the process
    if parent_element is not None and data_object_ref in parent_element:
        parent_element.remove(data_object_ref)

for data_store_ref in root.findall('.//{http://www.omg.org/spec/BPMN/20100524/MODEL}dataStoreReference'):
    parent_element = tree.find('.//{http://www.omg.org/spec/BPMN/20100524/MODEL}process')  # Assuming data stores are within the process
    if parent_element is not None and data_store_ref in parent_element:
        parent_element.remove(data_store_ref)

# Update sequence flows with the new references after replacements
for element in root.iter():
    if element.tag == "sequenceFlow":
        source_ref = element.get("sourceRef")
        target_ref = element.get("targetRef")
        if source_ref in element_mapping and target_ref in element_mapping:
            element.set("sourceRef", element_mapping[source_ref].get("id"))
            element.set("targetRef", element_mapping[target_ref].get("id"))

# Manually replace the namespaces in the root element's tag
root.tag = root.tag.replace('ns0:', 'bpmn:').replace('ns2:', 'bpmndi:').replace('ns3:', 'dc:').replace('ns4:', 'di:')

# Write the modified content to the output file
tree.write(input_file_name.replace(".bpmn", "_modified.bpmn"), encoding='utf-8', xml_declaration=True)
```

Figure A.2: Snippet of Code for generating Dual Partner

```
def process_zip_files(zip_file_name, output_zip_file_name):
    with zipfile.ZipFile(zip_file_name, 'r') as input_zip_file:
        with zipfile.ZipFile(output_zip_file_name, 'w') as output_zip_file:
            for file_name in input_zip_file.namelist():
                with input_zip_file.open(file_name) as input_file:
                    # Process each BPMN file in the zip
                    process_bpmn_file(input_file, output_zip_file)

if __name__ == "__main__":
    parser = argparse.ArgumentParser(description="Process BPMN files in a zip archive and generate new zip archive.")
    parser.add_argument("input_zip", help="Input zip file containing BPMN files")
    parser.add_argument("output_zip", help="Output zip file for modified BPMN files")
    args = parser.parse_args()

    process_zip_files(args.input_zip, args.output_zip)
```

Figure A.3: Code for batch input

```
import numpy as np
import pandas as pd

def augment_data(original_data, num_augmented_samples=5, time_variation_std=0.1, duration_variation_std=0.05):
    augmented_data = []

    # Convert timestamp column to pandas Timestamp
    original_data['timestamp'] = pd.to_datetime(original_data['timestamp'], format='%m/%d/%Y %H:%M')

    for _, row in original_data.iterrows():
        for _ in range(num_augmented_samples):
            # Time perturbation
            time_variation = np.random.normal(0, time_variation_std)
            augmented_timestamp = row['timestamp'] + pd.Timedelta(minutes=time_variation)

            # Duration perturbation
            duration_variation = np.random.normal(0, duration_variation_std)
            augmented_duration = row['duration'] + pd.Timedelta(minutes=duration_variation)

            # Convert Timedelta to string
            augmented_duration_str = str(augmented_duration)
```

Figure A.4: Code for the incorporation of dual partner with machine learning

```
            # Create an augmented sample
            augmented_sample = {
                'timestamp': augmented_timestamp,
                'order_id': row['order_id'],
                'customer_id': row['customer_id'],
                'product_id': row['product_id'],
                'activity': row['activity'],
                'decision': row['decision'],
                'outcome': row['outcome'],
                'duration': augmented_duration_str  # Use the string representation
            }

            augmented_data.append(augmented_sample)

    # Convert the augmented data to a DataFrame
    augmented_df = pd.DataFrame(augmented_data)

    return augmented_df
path = '/content/drive/MyDrive/data.csv'
#data =
original_data = pd.read_csv(path)
```

Figure A.5: Code for the incorporation of dual partner with machine learning :

```python
#Cleaning and feature engineering

def clean_and_engineer_features(data):
    # Data Cleaning
    # Handling Missing Values
    data = data.dropna()  # Remove rows with missing values
    # Ensure 'timestamp' column is of datetime type
    data['timestamp'] = pd.to_datetime(data['timestamp'], errors='coerce')
    # Convert 'duration' to numeric, handling non-numeric values
    data['duration'] = pd.to_numeric(data['duration'], errors='coerce')
    # Fill NaN values in 'duration' with the median
    data['duration'].fillna(data['duration'].median(), inplace=True)
    # Feature Engineering
    # Example: Extracting Date Components from Timestamp
    data['year'] = data['timestamp'].dt.year
    data['month'] = data['timestamp'].dt.month
    data['day'] = data['timestamp'].dt.day
    data['hour'] = data['timestamp'].dt.hour
    # Example: Creating a Binary Feature
    data['is_high_duration'] = np.where(data['duration'] > data['duration'].median(), 1, 0)
    # Example: Encoding Categorical Variables
    data = pd.get_dummies(data, columns=['activity', 'decision'])
    # Example: Scaling Numerical Features
    from sklearn.preprocessing import StandardScaler
    scaler = StandardScaler()
    numerical_columns = ['duration']
    data[numerical_columns] = scaler.fit_transform(data[numerical_columns])

    return data
```

Figure A.6: Code for the incorporation of dual partner with machine learning :

```python
# Simulation Model (Random Forest Regressor)
simulation_model = RandomForestRegressor(n_estimators=100, random_state=42)
simulation_model.fit(X_train, y_train)  # Fit the model with the training data
simulation_predictions = simulation_model.predict(X_test)
simulation_mse = mean_squared_error(y_test, simulation_predictions)
print(f"Simulation Model Mean Squared Error: {simulation_mse}")

# Optimization Model (Example: Minimize Mean Squared Error)
def objective_function(params):
    # Define the objective function to be minimized
    # Example: Mean Squared Error
    predictions = simulation_model.predict(X_train)
    mse = mean_squared_error(y_train, predictions)
    return mse

# Initial guess for optimization parameters
initial_params = [1.0, 0.5, 0.75]  # Replace with your actual initial parameters
result = minimize(objective_function, initial_params, method='Nelder-Mead')
optimized_params = result.x
print(f"Optimized Parameters: {optimized_params}")

# Reinforcement Learning Model (Example: Neural Network)
rl_model = models.Sequential([
    layers.Dense(64, activation='relu', input_shape=(X_train.shape[1],)),
    layers.Dense(1)
])
rl_model.compile(optimizer='adam', loss='mean_squared_error')
```

Figure A.7: Code for the incorporation of dual partner with machine learning :

```
# Evaluate the Reinforcement Learning Model
rl_loss = rl_model.evaluate(X_test, np.array(y_test,dtype=np.float32))
print(f"Reinforcement Learning Model Loss: {rl_loss}")


# Split the data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(features, target, test_size=0.2, random_state=42)


# Define the RandomForestClassifier
classifier = RandomForestClassifier()

# Define the hyperparameters to search through
param_grid = {
    'n_estimators': [50, 100, 200],
    'max_depth': [None, 10, 20],
    'min_samples_split': [2, 5, 10],
    'min_samples_leaf': [1, 2, 4]
}

# Use GridSearchCV to find the best hyperparameters
grid_search = GridSearchCV(classifier, param_grid, cv=5, scoring='accuracy')
grid_search.fit(X_train, y_train)
```

Figure A.8: Code for the incorporation of dual partner with machine learning

```
# Get the best parameters
best_params = grid_search.best_params_
best_params

# Get the best estimator (model)
best_model = grid_search.best_estimator_

# Get the best cross-validated accuracy
best_accuracy = grid_search.best_score_

# Print the results
print("Best Hyperparameters:", best_params)
print("Best Cross-validated Accuracy:", best_accuracy)

# Assuming X_test and y_test are your test data
predictions = best_model.predict(X_test)
test_accuracy = accuracy_score(y_test, predictions)

print("Test Accuracy:", test_accuracy)

# Fit the model with the best parameters
optimized_model = RandomForestClassifier(**best_params)
optimized_model.fit(X_train, y_train)
```

Figure A.9: Code for the incorporation of dual partner with machine learning

```python
# Create a mapping for dual partner replacements
dual_partner_mapping = {
    'activity_Order Placement': 'activity_Order Fulfillment',
    'activity_Payment Processing': 'decision_Approved',
    'decision_Completed': 'decision_Processing',
    # Add more mappings as needed
}

# Introduce columns for dual partner equivalents
for original_activity, dual_activity in dual_partner_mapping.items():
    df[dual_activity] = df[original_activity]

# Adjust the outcome based on replacements
for original_activity, dual_activity in dual_partner_mapping.items():
    df.loc[df[original_activity] == 1, 'outcome'] = df[df[dual_activity] == 1]['outcome']

# Drop original activity columns
df = df.drop(list(dual_partner_mapping.keys()), axis=1)

# Split the data into features (X) and target variable (y)
X = df.drop('outcome', axis=1)
y = df['outcome']

# Split the data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# Initialize the classifier (you can adjust hyperparameters)
classifier = RandomForestClassifier(n_estimators=100, random_state=42)
```

Figure A.10: Snippet of Code for generating Dual Partner

```python
# Iterate through each mapping and update the DataFrame
for original_activity, dual_activity in dual_partner_mapping.items():
    # Log the mapping
    log_data.append({
        'timestamp': datetime.datetime.now(),
        'original_activity': original_activity,
        'dual_activity': dual_activity,
        'action': 'dual_mapping'
    })

    # Update the DataFrame
    dual_mapping_df[dual_activity] = dual_mapping_df[original_activity]

    # Drop original activity columns
    dual_mapping_df = dual_mapping_df.drop(original_activity, axis=1)

# Log file DataFrame
log_df = pd.DataFrame(log_data)

# Save the dual-mapped dataset
dual_mapping_df.to_csv('dual_mapped_dataset.csv', index=False)

# Save the log file
log_df.to_csv('dual_mapping_log.csv', index=False)
```

Figure A.11: Snippet of Code for generating Dual Partner

References