INVESTIGATION OF MODELLING OF DYNAMIC BUSINESS PROCESSES

THESES OF THE PH.D. DISSERTATION
KHAWLA BOUFIA
SUPERVISOR: BáLINT MOLNÁR
THIS DISSERTATION IS SUBMITTED FOR THE FULFILMENT OF DOCTOR OF PHILOSOPHY IN COMPUTER SCIENCE

DEPARTMENT OF INFORMATION SYSTEMS
PH.D. SCHOOL OF COMPUTER SCIENCE
HEAD OF SCHOOL: PROF. ERZSÉBET CSUHAJ-VARJÚ
PH.D. PROGRAM OF INFORMATION SYSTEMS
HEAD OF PROGRAM: PROF. ANDRÁS ID.BENCZÚR

BUDAPEST, 2021
I would like to dedicate this thesis to my lovely family.
Declaration

I hereby declare that except where given reference is made to the work of others, the contents of this dissertation are original and have not been submitted in entire or in part for consideration for any other degree or qualification in this, or any other university.

This dissertation is my own work and contains nothing which is the outcome of work done in collaboration with others, except as specified in the text and acknowledgments.

May 2021
Acknowledgments

It was a very big challenge for me to finish this thesis. I begin a big thank you to GOD and his goodness for the patience, skill and encouragement he has given me to get to this point.

That is why I am very grateful to my supervisors Bálint Molnár for giving me the opportunity to do my Ph.D. in the information system department, for supporting me throughout all my studies, and for his invaluable help and advice, without which this work would not have been possible.

A sincere thank to my lovely parents Soumia and Derradji, and to my small sisters Malake, Inasse and Takwa, to my best brother Taki for their support and motivation during my studies and my life in general.

My colleagues in all computer science ELTE departments, and all my closest friends for their permanent support and encouragement throughout my studies.

Special thanks go to our Doctoral School and the EFOP project (The European Union, co-financed by the European Social Fund (EFOP-3.6.3-VEKOP-16-2017-00002) and (EFOP-3.6.3-VEKOP-16-2017-00001)) which supports my works and funded all my publications.

Finally, we would like to thank all those who, from near or far, have contributed to the completion of this work.
Abstract

The notion of the process has become an essential asset for daily life in organizations and enterprises. It helps achieve organizational objectives and defines information systems (IS) management. One of the most relevant problems in IS research concerns business alignment with the change of the environment, customer’s needs, and requirements. The alignment process was attempted to be integrated into several domains to achieve particular goals and desired outcomes for a long time. When a process’s objective relates to the company’s goals, the process will be designated as a Business Process (BP).

The BPs are interrelated to each other considering their structure and functionalities which define their static and dynamic aspects. Most research efforts concentrated on static BP problems, i.e., on the process structure. Still, the traditional approach used to implement a BP in today’s IS no longer covers the dynamically changing business environment that is the business’s actual need. Because of that, this Ph.D. dissertation intends to widely cover BP concepts in general, with a particular interest in its dynamic aspect introducing a fresh, and new perspective that defines the proposed approach clearly.

Previous studies dealt with only just mentioning the concept of the dynamic aspect of processes, and some related terms were described but without specific definition, researchers in the same domain as well did not define the different factors that impact it and their components during BP execution (run-time). Our study contributes to the domain of BP by providing its clear and comprehensive definition and the conceptualization of its dynamic aspect. The latter helps in familiarizing with BP functionalities during the run-time, and we also investigated factors influencing it and their change as well as elements or objects impacted by these changes. While
some researchers concentrated on the structure, although they neglected when BP
elements changed and how they were modified. For that reason, we deeply investigated
that area, we have identified the various elements (components) that can be affected
by the factors of change and how the process adapts to this influence.

BP modeling describes the functioning of a process and defines all activities of the
BP model for giving more understanding. The thesis establishes a taxonomy for the
negative and positive features (advantages and disadvantages) for existing BP models.
These similarities have helped us find syntactic code elements in the representation
codes and functionalities that proved helpful in transformations between models.
The field of transformations between models is a vast area, and a lot of methods for
conversions were described but never between models chosen by us (BP Execution
Language (BPEL) and Finite State Machines (FSM)), and (FSM and Hypergraph).
The comparison across models may also lead to mixing and integrating different
models to find a way to verify formally unapproved ones.

We have introduced the hypergraph concept in this modeling area that enables
the utilization of graph algorithms, linear algebra, and the most current data science
methods. This operationalized model supports the model checking after incorporating
changes into the running instances of dynamic processes. We implemented some
representations dealing with this concept with various matrices and forms. In our
future works, we can use hypergraph and its models for either different models of
processes or apply several new verification methods of the soundness of processes
that were changed in a dynamic way. The implementation of our approach helps to
use many tools to verify the process and check some properties through exploiting
the hypergraph representation.
## Contents

1 Introduction .......................................................... 1
   1.1 Motivation ...................................................... 1
   1.2 Research Problem ............................................. 3
   1.3 Theses .......................................................... 4
   1.4 Scientific Contribution ........................................ 6
   1.5 Publications .................................................. 8
   1.6 Dissertation Organization and Structure ..................... 13

2 Literature Review .................................................... 15
   2.1 Enterprise and Information systems .......................... 15
   2.2 Business Processes Theory ................................... 16
   2.3 Business Processes ............................................ 19
      2.3.1 Process Notion Definition ............................. 19
      2.3.2 Business Process Definition .......................... 21
      2.3.3 Business Process Management ......................... 24
      2.3.4 Business Process Life-Cycle ......................... 24
   2.4 Modelling of Business Processes ............................ 27
      2.4.1 BP Model Definition .................................... 27
      2.4.2 Usefulness of Business Process Models ............... 28
      2.4.3 Business Processes Representation (Models) .......... 29
   2.5 Business Process Technologies .............................. 30
      2.5.1 The Concept of Workflow .............................. 30
      2.5.2 Workflow Management Systems ......................... 31
      2.5.3 The Role of Workflow Engine .......................... 31
      2.5.4 Enterprise Architecture (EA) .......................... 32
CONTENTS

2.5.5 Web Services .............................................. 38

3 Business Process Models ........................................ 41
  3.1 Existing Business Process Models ............................ 41
     3.1.1 Petri-nets ............................................. 42
     3.1.2 Finite State Machines (FSM) .......................... 43
     3.1.3 Business Process Model Notation (BPMN) .............. 47
     3.1.4 Unified Modelling Language (UML) .................... 48
     3.1.5 Business Process Execution Language (BPEL) ........... 49
     3.1.6 Process Algebras ...................................... 50
     3.1.7 Event Driven Process Chain (EPC) .................... 51
  3.2 Comparison of Business Process Models ...................... 52
  3.3 BP Models Advantages and Disadvantages ................... 54
  3.4 Models Comparison ........................................... 56
  3.5 Model Transformations ....................................... 58
  3.6 Functional Approach To Transforming Abstract Specifications of FSM from BPEL Program ......................... 60
     3.6.1 The Pattern-based Approach for transforming BPEL Program to FSM ............................................. 62
     3.6.2 Presentation of the Working Environment .............. 63
     3.6.3 System Architecture .................................. 64
     3.6.4 System Functionalities ................................ 65
  3.7 Hypergraph Representation Based on FSM ..................... 68

4 Dynamic Business Process Investigation ......................... 70
  4.1 Dynamic Business Processes .................................. 70
  4.2 Dynamic Business Process Management ....................... 73
  4.3 Factors Impacting Dynamic BP ................................ 74
  4.4 Process Components that Change at Run-time ................ 76
  4.5 Dynamic BP Requirements .................................... 78
  4.6 Limitations Exist in Current Research ....................... 81
  4.7 Measures Related to Dynamic BP Quantification ............. 81
  4.8 Facilities for Quantification and Modelling Dynamic BP? .. 83

5 Hypergraphs and Formal Verification ............................ 86
# List of Figures

<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.1</td>
<td>Travel Reservation</td>
<td>20</td>
</tr>
<tr>
<td>2.2</td>
<td>Business Process Management Life-Cycle.</td>
<td>25</td>
</tr>
<tr>
<td>2.3</td>
<td>Enterprise Architecture Components</td>
<td>34</td>
</tr>
<tr>
<td>2.4</td>
<td>Zachman Framework</td>
<td>37</td>
</tr>
<tr>
<td>2.5</td>
<td>TOGAF Architecture Domain Structure</td>
<td>38</td>
</tr>
<tr>
<td>3.1</td>
<td>An Example of Petri-nets</td>
<td>43</td>
</tr>
<tr>
<td>3.2</td>
<td>An Example of FSM structure</td>
<td>44</td>
</tr>
<tr>
<td>3.3</td>
<td>Example of Sequence Structure in FSM</td>
<td>45</td>
</tr>
<tr>
<td>3.4</td>
<td>Example of Multiple Choice Structure in FSM</td>
<td>45</td>
</tr>
<tr>
<td>3.5</td>
<td>Example of Parallel Split Structure in FSM</td>
<td>46</td>
</tr>
<tr>
<td>3.6</td>
<td>Example of Multiple Merge Structure in FSM</td>
<td>46</td>
</tr>
<tr>
<td>3.7</td>
<td>Example of Cycle structure in FSM</td>
<td>47</td>
</tr>
<tr>
<td>3.8</td>
<td>BPMN Diagram of iPhone Ordering Process from eBay</td>
<td>48</td>
</tr>
<tr>
<td>3.9</td>
<td>An Example of Document Management Process Activity</td>
<td>49</td>
</tr>
<tr>
<td>3.10</td>
<td>Example of EPC</td>
<td>51</td>
</tr>
<tr>
<td>3.11</td>
<td>Model Transformation</td>
<td>58</td>
</tr>
<tr>
<td>3.12</td>
<td>System Architecture</td>
<td>66</td>
</tr>
<tr>
<td>4.1</td>
<td>The Architectural Building Blocks for Unified Modelling Framework</td>
<td>77</td>
</tr>
<tr>
<td>5.1</td>
<td>Hypergraph Structure</td>
<td>87</td>
</tr>
<tr>
<td>5.2</td>
<td>Hypergraph Example</td>
<td>88</td>
</tr>
<tr>
<td>5.3</td>
<td>AWS Services</td>
<td>92</td>
</tr>
<tr>
<td>5.4</td>
<td>AWS CloudWatch</td>
<td>93</td>
</tr>
<tr>
<td>5.5</td>
<td>Approaches Schema</td>
<td>94</td>
</tr>
</tbody>
</table>
LIST OF FIGURES

5.6 Third Approach Schema. ................................................. 95
5.7 Generate Simulated BP Data. ........................................... 96
5.8 From Bipartite Graph to Hypergraph. ............................... 96
5.9 Smith Normal Form. ...................................................... 97
5.10 Principal of Model Checker. ........................................... 100
List of Tables

2.1 The Most Established EA Frameworks (Development and types) . . . 36
3.1 Comparative Analysis Of BP Models . . . . . . . . . . . . . . . . . 54
3.2 BP Models Advantages and Disadvantages . . . . . . . . . . . . . 56
3.3 Existing Transformations of FSM and BPEL . . . . . . . . . . . . . 62
4.1 Definitions related with dynamic BP . . . . . . . . . . . . . . . . . 73
4.2 Dynamic BP Requirements and their Roles . . . . . . . . . . . . . 80
List of Abbreviations

ACP  Algebra of Communicating Processes
ADM  Architecture Development Method
AD   Activity Diagrams
AHL  Algebraic High-Level
API  Application Programming Interface
AUML Agent Unified modelling Language
AWS  Amazon Web Services
B-arc Backward Hyperarcs
BDD  Binary Decision Diagrams
BF-hypergraph Forward-Backward Graph
BPEL Business Process Execution Language
BPE  Business Process Engine
BPMI Business Process Management Initiative
BPMN Business Process Model Notation
BPMS Business Process Management Systems
BPM Business process Management
BPR  Business Process Re-engineering
BP  Business Process
LIST OF ABBREVIATIONS

**C4ISR**  Command, Control, Communications, Computers, Intelligence, Surveillance and Reconnaissance

**CCS**  Calculus of Communicating Systems

**CEP**  Complex Event Processing

**CSP**  Communicating Sequential Processes

**CTL**  Computation Tree Logic

**DEVS**  Discrete Event System Specification

**DYPROTO**  Tools for Dynamic Business Process

**EAM**  Enterprise Architecture Management

**EA**  Enterprise Architecture

**EPC**  Event Driven Process Chain

**ERP**  Enterprise Resource Planning

**F-arc**  Forward Hyperarc

**FSM**  Finite State Machines

**GSPN**  Generalized Stochastic Petri Nets

**HOL**  higher-order logic

**HTTP**  Hypertext Transfer Protocol

**IDE**  Integrated Development Environment

**IPMS**  Performance Measurement Systems

**IS**  Information Systems

**IT**  Information Technology

**JADE**  Agent DEvelopment framework

**JAR**  Java ARchive

**JDBC**  Stands for Java Database Connectivity

**JDOM**  Java Document Object Model
LIST OF ABBREVIATIONS

LTL  Linear Temporal Logic
MDD  Model-Driven Development
NIST National Institute of Standards and Technology
OMG  Object Management Group
OO   Object-Oriented
PHP  Hypertext Preprocessor
PID  Principal Ideal Domain
QVT  Query/Views/Transformation
RBPMo Role-Driven Business Process modelling
S-GNet ScriptGen technology
SBPL  Standard Business Process Language
SM   State Machine
SOAP Simple Object Access Protocol
SOA  Service Oriented Architecture
SOS  Structural Operational Semantic
TAFIM Technical Architecture Framework for Information Management
TPGAF ThE Open Group Architecture Framework
UDDI  Universal Description, Discovery, and Integration
UML-SM UML State Machines
UML  Unified modelling Language
URI  Uniform Resource Identifier
W3C  World Wide Web Consortium
WfMC Workflow Management Coalition
WfMS Workflow Management System
LIST OF ABBREVIATIONS

**WSDL**  Web Services Description Language

**XML**  Extensible Markup Language

**XOR**  Exclusive Or

**YAWL**  Yet Another Workflow Language
Chapter 1

Introduction

There is a plethora of studies about IS, BP and various known researchers who have diverse viewpoints in this domain.

The introduction chapter aims to provide a background for all the research details that are to follow in the subsequent parts of the study. It initiates the context of the topic and focuses overview emphasizing the importance of the conducted research problem.

In my master thesis, my interest was in the notion of BP and its management. Our overall goal in this thesis is to increase knowledge regarding the concept of BP, its modelling, and its management with a deep investigation especially on the process aspects (static and dynamic).

1.1 Motivation

In this section, I wanted to shed light on the motives behind drafting these research questions that attempted to answer. It will discuss and mention all causes of why we are investigating the dynamic aspects of BP, its modelling, and its management.

While several research [58] [112] [103] [121] have been carried out in the field of static BP, the dynamic aspect is very limited in research. So far, there are no studies that focus on it. Most of the researchers concentrate on the structure of BP but not on its functionalities during the run-time (execution time), in addition, there
are no publications discussing the different changes or factors impacting BP during that time. The scarcity of articles in this perspective hinders our understanding of organization adaptation to these types of factors. These various reasons are the motives behind investigating this topic.

In recent years, the term "BP management (BPM) and modelling" [43] became widely known and plays a pivotal role in most organizations to adapt to new business philosophy and applied informatics. This has led to the emergence of a large number of BP modelling methods and tools for the elapsed decades. Methods for the business modelling process could be regarded as a description by which people and stakeholders of the enterprise can understand the object, documents, systems, and related phenomena.

Various process modelling methods are fundamental to describe the structure and the functioning of the company. In our research, we emphasize on BP models and existing representations, as we intend to compare these models based on some criteria (semantic, syntactic, and structural) to establish differences and similarities which help in transformations among similar ones. These transformations also aid to make combinations and integration between different models to find a way or tools (such as formal verification tools) to verify formally some models which might not be verified.

Our focus available BP models are to define business requirements and propose a BP model structure for flexible visualization of BP models and ensure more comprehensive business requirements. Additionally, these BP technologies differ in their requirements and their use contribute to building dedicated global information about functionalities, rules, and characteristics for supply and manage dynamic BP (the execution of the process during run-time) based on the concept of the process model (representation).

Existing approaches dealing with dynamic BP modelling and simulation are incomplete because they are lacking in terms of theory and practice (implementation). Furthermore, there is no commonly affirmed definition of dynamic BP, the same concern is noticed with the definition of dynamic BP, for that, our main goal is to deeply investigate the several existing and famous references in the domain to conclude with the most appropriate and our own definitions for the concept.
One of the other motives is related to the use of hypergraphs [22], they are a widely used graph-based formalism introduced for the modeling of various kinds of systems. The formalism of hypergraph is quite versatile and its main strength is its built-in capability of succinctly representing relationships between the unbounded number of nodes. A relatively lengthy section will be dedicated to hypergraph representation and code generation allowing us to describe it in different forms. This will enable us to use it in an advanced stage to represent BP and solve many sensitive types of BP such as health BP and financial (economical) enterprise affected recently by the world situation (Covid-19).

1.2 Research Problem

To adapt IS to the versatility of the environment, organizations invest large budgets and efforts in managing the evolution of BP. However, one of the major encountered difficulties is the development of techniques and tools that deal with these changes. Indeed, in the context of dynamic evolution, the continuity of active instances, following the new process is a crucial problem that requires an analysis of the impact of changes in BP on active instances.

Changes in BP can arise because of several reasons including environmental changes, maintenance of management rules and alterations in the organization structure, the optimization of BP. As an immediate consequence of these changes, companies must continually renew and improve their BP to acclimate to the developments. To deal with the inherent changes in BP, we should focus on both static and dynamic aspects of BP, the static one means there are no instances of this process running. In this case, the changes will take effect only for new instances and they will not generally have any impact regarding the already completed instances.

For the dynamic one, there arises the crucial question-related management of active instances of a BP whose specification has been modified. For that, we are required to be more interested in the exploitation of BP concepts to adopt this technology. We thoroughly investigate existing BP functionalities and the various changes which can impact them during the run-time. We also deal with the various tools and techniques allowing both modelling and automation of the execution of the BP in an IS using different existing representations (models).
Based on expected outcomes and research objectives, the proposed research attempts to answer research questions summarized as follows:

- **Question 1**: What are the similarities, differences, advantages, and disadvantages between existing BP models?
- **Question 2**: How can we properly define the dynamic BP concept and its modelling?
- **Question 3**: What are the factors influencing BP functionalities (during running time)? Moreover, which components could be impacted?
- **Question 4**: How can BP dynamicity be ensured, and what requirements of dynamic BP must be considered?
- **Question 5**: In the future, and after learning about the different changes that could impact it, can the organization adapt and control the different changing factors?

These questions will enable us to investigate the topic of dynamic BP and discuss the different problems that arise for detailed analysis of the domain. Research objectives were broken down into analyzable research questions and the main research objective strongly connects to the problem statement formed in the previous subsection.

### 1.3 Theses

- **Thesis 1**: An adequate framework for the analysis of the research domain and understanding the various concepts related to the BP is defined that laid down the foundation of further investigation.

  We investigated the field in general and collect the most essential research details [68], definitions, terms, standards for IS, BP modelling, BPM and BP technologies such as Enterprise Architecture (EA) [18], Web services [17], as well as we presented the relation between those all cited notions, Our overall goal in this part is to increase knowledge regarding the concept of BP, its management. All those definitions found in chapter 2 and details help us to know more the research domain and facilitate later their use.
• **Thesis 2:** in this dissertation, a taxonomy is created for comparing the various existing process models. The taxonomy constitutes the theoretical ground of model transformation.

We presented the most BP models and established taxonomy for the negative and positive features [82] (advantages and disadvantages) for each existing model. A comparative analysis [89] part were summarized in details in tables a comparison which based on various criteria (syntax, structure, and semantic), these similarities concluded aide us to find syntactical element in the representation codes and functionalities that proved helpful in transformations between models (transformations have done: [67] [17]). The comparison across models may also lead to mixing and integrating different models to find a way to formally verify ones that might not be verified. At the end of this part, we can say that the first question of our investigation answered.

• **Thesis 3:** in this part, we discussed and answered that most questions posed. We investigated deeply the dynamic BP aspect definitions [65] using several references in the same domain. Our own definition of dynamic BP is created, which is one of the important goals of our study.

The several factors influencing BP [69] during its functioning were well defined in chapter 3, we concluded them using a collection of information from various references, later the elements or objects impacted by these changes were explained. The question posed about requirements needed that must take into consideration for dynamic BP also answered in that part. we discuss as well how to ensure BP dynamically and how it should be managed [63]. At this thesis, a big part of our investigation done.

• **Thesis 4:** this thesis part introduces the hypergraph that enables the utilization of graph algorithms along with the most recent methods of Data Science. The notions (definitions and concepts applied to). An implementation done of hypergraph was well-explained. An implementation of the hypergraph concept [82] has done in Python, this representation using matrices and forms can be used in our future works to use hypergraph and its instances for either other processes and apply several new methods, the implementation aims to represent the hypergraph in different ways to easier the use of it later.
Formal Methods and formal verification objectives are defined with particular attention to model checking, which applied on the representation of the hypergraph based on FSM approach [67] and verify several properties [66].

1.4 Scientific Contribution

In line with the problems outlined, I contributed towards the domain of BP in general by:

- Investigating in the domain, defining the majority of concepts [18] [65] related and explain the relation between them. There is a multitude of tools with varying sets of features for the modeling, we present the different existing BP models in details based on several criteria (semantic, syntactic, and structural) [89] and well defined their advantages and disadvantages [82],

The comparison made contributed to finding some similarities and differences between such kinds of models which helped as well to search for not existing transformations between models and aided us to collect more ideas about those deals and made transformations and combinations between them.

During our study there transformations were done, the first is a new functional approach that transforming abstract specifications based on FSM from BPEL Programs. The application that we have implemented [17] contains several functionalities allowing us to perform several tasks. It revolves around the transformation rules which enable the construction of a moving structure between the program written in BPEL language based on patterns.

The proposed approach has been implemented in an environment that concertizes the implementation of patterns. The developed software tool also performs functions beneficial for protocol managers, such as BPEL code generators, verification, and visualization of some FSM properties and defining abstraction levels (model instead of language).

The second one [67], was a proposed approach that aims to describe the hypergraph to model and all its elements and patterns based on the FSM concept in order to use a formal method on hypergraph to verify such BP properties which might never be verified before with formal method.
CHAPTER 1. INTRODUCTION

All transformations mentioned before were never done only by us which contributed in the domain of transformations models and augment the possibility of sharing the verification method of several properties especially formal methods.

• We investigating both aspects of the BP static (the structure of BP) and the dynamic (functionalities of BP), because the researchers focus more on the static aspects, we are especially focused more on the dynamic one and enriching this domain by adding and give a new specific definition give a specific definition of this concept [69] (providing our own definition);

• The changes affecting BP during execution time were defined after a deep investigation, while some researchers were concentrated on the structure but neglected why BP elements changed. we listed the factors that might impact the functionalities and specify which components [69] can be affected or influenced by those changes factors.

  We focused on that and define different items that could be affected or influenced. addition to that, the functionalities that can be broken, stopped, or modified will be more specified to avoid them in the future and minimize their impact with new proposal approaches;

• The modelling of BP using different models gives BP easier management. In our studies as well we will try to represent the concept of a hypergraph and implement some representations with different forms.

  The representation can generally help in using many verification tools (formal verification tools) of the process, the different ways used to represent the hypergraph as matrices and forms which aide to manipulate it more, that combination which never done between hypergraph with other formal models like FSM helped us to check different properties of the process using a formal tool named model checker [66];

• Hypergraph representation allowing using it in a later stage to represent BP and solve new sensitive kinds of BP which are especially affected by the world contemporary situation.
1.5 Publications

This section summarises the list of my publications, in chronological order:

- **Publication 1:**
  
  - **Authors:** Molnár Bálint and Bouafia Khawla.
  
  - **Title:** Adaptive Case Management and Dynamic Business Process modelling A proposal for document-centric and formal approach.
  
  - **Conference:** The 12\textsuperscript{nd} International Symposium on Applied Informatics and Related Areas (AIS 2017), November 9, 2017, Székesfehérvár, Hungary.
  
  - **Published in:** AIS 2020 Proceedings.
  
  
  - **Author contributions:** Khawla Bouafia and Bálint Molnár worked on this paper together; we write the original draft version, then we carried out editing the revision. Bálint Molnár proofread the draft and revision, supervised the process.

- **Publication 2:**
  
  - **Authors:** Bouafia Khawla and Molnár Bálint.
  
  - **Title:** An FSM Approach for Hypergraph Extraction Based on Business Process modelling.
  
  - **Conference:** The 3\textsuperscript{rd} International Conference on Computer Science and its Applications (CSA 2018), April 2018, Algiers, Algeria.
  
  - **Published in:** Lecture Notes in Networks and Systems, vol 50. Springer, Cham.
  
  - **Link:** [https://link.springer.com/chapter/10.1007/978-3-319-98352-3_17](https://link.springer.com/chapter/10.1007/978-3-319-98352-3_17).
  
  - **Author contributions:** Khawla Bouafia and Bálint Molnár worked on this paper together; we write the original draft version, then we carried out editing the revision. Bálint Molnár proofread the draft and revision, supervised the process.
CHAPTER 1. INTRODUCTION

• Publication 3:
  • Authors: Bouafia Khawla and Molnár Bálint.
  • Title: Dynamic Business Process: Comparative Models and Workflow patterns.
  • Conference: Proceedings of the conference.
  • Link: http://acta.bibl.u-szeged.hu/id/eprint/61760.
  • Author contributions: Khawla Bouafia and Bálint Molnár worked on this paper together; we write the original draft version, then we carried out editing the revision. Bálint Molnár proofread the draft and revision, supervised the process.

• Publication 4:
  • Authors: Bouafia Khawla and Molnár Bálint.
  • Title: A Survey on Dynamic Business Processes and Dynamic Business Processes Modelling.
  • Link: https://www.scitepress.org/Link.aspx?doi=10.5220/0007627105560563
  • Author contributions: Khawla Bouafia and Bálint Molnár worked on this paper together; we write the original draft version, then we carried out editing the revision. Bálint Molnár proofread the draft and revision, supervised the process.

• Publication 5:
  • Authors: Bouafia, Khawla and Molnár, Bálint.
CHAPTER 1. INTRODUCTION

• **Title:** Opportunities for Supporting Digital transformation through Modelling Dynamic Processes.

• **Conference:** The 21\textsuperscript{st} International Conference on Enterprise Information Systems (ICEIS 2019), 2019, Heraklion, Crete, Greece.

• **Published in:** Proceedings of the 21\textsuperscript{st} DCEIS, 3-9, 2019, Heraklion, Crete, Greece.

• **Link:** [https://heptagon.in/2019/08/14/digital-transformation-through-dynamic-business-processes-modeling/](https://heptagon.in/2019/08/14/digital-transformation-through-dynamic-business-processes-modeling/)

• **Author contributions**: Khawla Bouafia and Bálint Molnár worked on this paper together; we write the original draft version, then we carried out editing the revision. Bálint Molnár proofread the draft and revision, supervised the process.

• **Publication 6:**

  • **Authors:** Bouafia Khawla and Molnár Bálint.

  • **Title:** Analysis Approach for Enterprise Information Systems Architecture Based on Hypergraph to Aligned Business Process Requirements.

  • **Conference:** CENTERIS-International Conference on ENTERprise Information Systems. November 2019, Sousse, Tunisie.

  • **Published in:** Procedia Computer Science Volume 164, 19-24. Published by Elsevier B.V. (counted as journal paper)

  • **Link:** [https://www.sciencedirect.com/science/article/pii/S187705091932188X](https://www.sciencedirect.com/science/article/pii/S187705091932188X)

  • **Author contributions**: Khawla Bouafia and Bálint Molnár worked on this paper together; we write the original draft version, then we carried out editing the revision. Bálint Molnár proofread the draft and revision, supervised the process.

• **Publication 7:**

  • **Authors:** Bouafia Khawla, Ali Khebizi and Molnár Bálint.
CHAPTER 1. INTRODUCTION

- **Title:** New Functional Approach to Transforming Abstract Specifications of Web Services Based on FSM from BPEL Programs\(^1\).


- **Link:** https://american-jiras.com/Abstract_3_April_2020.html

- **Author contributions:** Khawla Bouafia worked on this paper together on the idea of Ali Khebizi under the supervision of Molnár Bálint; we write the original draft version, then we carried out editing the revision. Khawla proofread the draft and revision.

- **Publication 8:**
  - **Authors:** Bouafia Khawla, Maxim Kumundzhiev and Molnár Bálint.
  
  - **Title:** Application of Models and Hypergraphs on Dynamic Aspects of Business Process Performance Analysis.

  - **Conference:** Zoltán Horváth (Horváth Zoltán Matematika-informatika) ELTE/ELU FoI/ICS/ELTE IK Department of Programming Languages And Compilers ; Adrian Petrușel (eds.) Collection of Abstracts : 13\(^{th}\) Joint Conference on Mathematics and Informatics Conference: Budapest, Hungary 2020.10.01. - 2020.10.03. (Babeș-Bolyai University of Cluj-Napoca, ELTE Faculty of Informatics; Eötvös Loránd Tudományegyetem Informatikai Kar, ELTE Informatikai Kar, Eötvös Loránd University). Budapest: ELTE Faculty of Informatics; Eötvös Loránd Tudományegyetem Informatikai Kar, Universitatea Babeș-Bolyai UBB, Babeș-Bolyai Tudományegyetem, Matematika és Informatika Kar, Babeș-Bolyai University of Cluj-Napoca, pp 37-37 (2020).Language: English

  - **Published in:** Conference paper ( Abstract ) | Scientific | RIS-Egyéb Integrált kutatói utánpótlás-képzési program az informatika és számítástudomány diszciplináris te...(EFOP-3.6.3-VEKOP-16-2017-00002) Funder: EFOP-VEKOP, TKP2020-NKA-06(Thematic Excellence Programme TKP2020-NKA-06 (National Challenges Subprogramme)) Funder: NKFI Last Modified: 2021.02.03. 11:27 Tibor Szabó (ELTE IK admin 5).  

\(^1\)This work will be extended with another part soon

Author contributions: Khawla Bouafia and Maxim Kumundzhiev worked on this paper on the idea together under the supervision of Molnár Bálint; we write the original draft version, then we carried out editing the revision. Molnár Bálint proofread the draft and revision.

• **Publication 9:**

  • **Authors:** Bouafia Khawla and Molnár Bálint.
  
  • **Title:** Formal Verification of Analysis Approach for Enterprise Information Systems Architecture Using Hypergraph Representation Based on Finite State Machines for Supporting Business Process Requirements.
  
  • **Journal:** Journal of Applied Business and Economics (JABE), (accepted under publishing). Vol.22(9), 03-01-2021.
  
  
  • **Author contributions:** Khawla Bouafia and Bálint Molnár worked on this paper together; we write the original draft version, then we carried out editing the revision. Bálint Molnár proofread the draft and revision, supervised the process.

• **Publication 10:**

  • **Authors:** Jozef Stašák, Jaroslav Kultan, Peter Schmidt and Bouafia Khawla.
  
  • **Chapter Title:** Business Process Linguistic Modelling –Theory and Practice - Part II: BPLM Business Process Designer Process Requirements
  
  • **Book:** Open Access book, "Operations Management - Emerging Trend in the Digital Era" edited by Dr. Antonella Petrillo (waiting for the online publication of the chapter )
  
  
  • **Author contributions:** Khawla Bouafia contributed in dynamic BP parts in the paper, Jozef Stašák proofread the draft and revision, supervised the process.
CHAPTER 1. INTRODUCTION

Submitted Journals

• 1. Authors: Bouafia Khawla, Maxim Kumundzhiev and Molnár Bálint.

• Title: Application of Models and Hypergraphs on Dynamic Aspects of Business Process Performance Analysis.

• Journal: Under journals’ standard reviewing (Special issues of journals (Annales)). Author contributions: Khawla Bouafia and Maxim Kumundzhiev worked on this paper on the idea together under the supervision of Molnár Bálint; we write the original draft version, then we carried out editing the revision. Molnár Bálint proofread the draft and revision.

• 2. Authors: Khawla Bouafia and Molnár Bálint.

item Title: Hypergraph Application on Business Process

• Journal: Journal of MDPI (submitted 4 May 2021)

• Author contributions: Khawla Bouafia and Bálint Molnár worked on this paper together; we write the original draft version, then we carried out editing the revision. Bálint Molnár proofread the draft and revision, supervised the process.

1.6 Dissertation Organization and Structure

The manuscript is composed of five chapters, the present preamble chapter is general introduction that specifies the context of the study, the motives behind the research, thesis contributions, my set of publications and the organization of the manuscript.

• In the first chapter 1, we provide a general idea as an introduction to our Ph.D. dissertation, motives, the contribution of this and the list of my publications.

• In chapter 2: We present the literature review and briefly recall the definitions, concepts, and standards for IS, BP and BP technologies. The focus will be particularly on the concept of BP, its modelling.

• In chapter 3: This chapter shows all concepts and research about BP models and various studies investigated with these models to insensitively compare between them and to point out the main differences between models (negative
CHAPTER 1. INTRODUCTION

...and positive points) as well as the model transformations made during our studies.

- In chapter 4: This chapter aims to report on a study that looked thoroughly at the dynamic aspect of BP, it will further discuss the changes that impact the process during its functioning time and their adjustment.

- In chapter 5: In this chapter, our focus is on the hypergraph representation, its theory and the various application domains, a hypergraph representation that we created using Python will be represented to facilitate some tasks. Several definitions and terms related to formal verification will be later used to verify our proposal approach of hypergraph based on FSM representation.

- Finally, the general conclusion given in chapter 6 summarizes the contributions of the thesis joined with the outlines and prospects for future research.

- At the end of the dissertation, appendices and references are listed.
Chapter 2

Literature Review

Introduction

This Ph.D. dissertation establishes the theoretical background that summaries definitions of the used concepts in our research domain. The purpose of the literature review is to assess the state of the art in the research focus being investigated and It surveys the contributions of existing literature to the dissertation topic.

The literature review aims to outline existing knowledge on the research field and presents the milestones of constituting research directions; i.e. to demonstrate how far advanced finding researchers in the same research topic and what they did in the same domain and allows also the researcher to identify if there is any gap [137] in it.

This chapter is dedicated to all concepts related to the term BP and its modelling and management. In our research, we intend to shed light on the term BP and investigate some fundamental concepts related to it. We start the chapter with the enterprise and IS concepts and the theory of BP, then we define BP, present the basic terms associated with it. Further.

2.1 Enterprise and Information systems

IS represents a major investment for any firm in today’s business environment to aspiring to achieve its business goals. For this reason, it should support organization
systems that represent fundamental elements of a business (people work process tasks structure and control system) during any decision. Indeed, the notion of "process" plays a primordial role in the control of IS evolution and associated Information technology (IT) systems. They are often considered as important to the design of dynamics of an organization IS [25] and this by providing robust processes, and adapted to their activities.

The company is a system that aims to create added value for its internal and external partners and interlocutors where information is a vital resource for its work. This added value is obtained by coordinating various activities that are carried out using the actors and resources of the company. In this context, it is interesting to know how training systems function within organizations.

While the first three (hardware, software, data,), fitting under the technology category, the remaining last two components, people and process are what separate the idea of IS from more technical fields, such as computer science. The components of IS reveal it crucially within an organization that can be expressed by its components, one of its most important additions is to convert data into information which transformed into organizational knowledge. After the definition of the IS and its components, BP theory will be discussed in the next section.

### 2.2 Business Processes Theory

Two hundred years ago Adam Smith\(^1\) introduced his ideas that industrial work should be broken down into tasks which could be sequentially conducted by different people i.e.BP. This process idea was refined with "Scientific Management" [122] by Taylor, which aims to describe processes in high granularity to increase efficiency. Shewhart [115] introduced "statistical process control" as a means to understand and improve processes. An important early description of processes was that of economist Adam Smith in his famous example of a pin factory inspired by an article in Diderot’s Encyclopedia.

In contrast to Smith’s view, which was limited to the same functional domain

---

and comprised activities that are in direct sequence in the manufacturing process, today’s process concept incorporates an important characteristic of cross-functionality. Following his ideas, the division of labor was widely adopted, while the integration of tasks into a functional, or cross-functional process was not considered as an alternative option until much later[113].

Several definitions of the process are usually short and succinct. Ould’s book BP [93] had still not attempted to define the term ‘BP’ by the end of the first chapter. Similarly, Jacobson [61] as well describes it without any explication, Bider [13] suggests that the re-engineering (BPR) community feel there is no great mystery about what a process is? They follow the most general definition of BP proposed by Hammer and Champy [54] that a process is a ‘set of partially ordered activities intended to reach a goal’.

Communities began when individuals specialized to trade or sell their skills and wares in exchange for other products or currency. Because companies evolved is no longer adequate for merely offering their goods for sale and, in order to maintain their work, they had to keep their competitive advantage. In the ’60s industry concentrated on how to produce more (quantity), after in the ’70s how to produce it cheaper (cost), then in the ’80s how to produce it better (quality) and how to produce it quicker (lead time). In the ’90s, finally in the 21st century how to offer more (service).

A production process was seen as a progression of taking raw material and transforming it into product. Activities were studied, partitioned, standardized and those activities conducive to automation were transferred to machine production. Activities too complicated or diverse remained in the hands of the human operators.

The characteristics of the processes studied and the descriptions proposed are, however, more indicative of production processes than a generic description of a BP. This is not to say they do not have value. Their ‘limited’ view of a process is a substantial area of research in itself. These techniques have been used in many fields from work-study and operations management, process control, business modelling, and systems engineering to name but a few, and are central BPR and BPM. The quality of a product (or service) offered by the company and the process that led to its development are closely linked. The concept of the process has long been integrated into industry sectors and is gradually becoming a major issue in other
CHAPTER 2. LITERATURE REVIEW

areas.

Processes are considered a generic factor in all organizations, it is viewed as strategic assets, which require companies to take a BP orientation. A plethora of research streams exists which is closely related to the concept of process [106]. They argue strongly that data must be collected against BP which is about how the business is organized. BPM in this context considers the process as both a business imperative and a means of understanding and explaining business activities, the way customer requirements are transformed into actual goods and services.

There has been considerable interest in the opportunities [68] to automate BP, this encompasses both the development of executable process management technology, as well as tools for modelling and designing processes.

Various companies and organizations report their successful experiences by applying revolutionary approaches to obtain dramatic, radical, and fundamental changes as Hammer and Champy [54] suggested. However, people rethought the myths of BPR after recognizing that 70 percent of BPR’s efforts failed [33]. One of these myths is that BP redesign does not come from a clean slate to build new processes from scratch. Notably, BP Modelling is essential within a BPR life-cycle. The BP Modelling within BPR mainly plays two important roles:

1. Capture existing processes by structurally representing their activities and related elements;

2. Represent new processes to evaluate their performance.

Curtis [28] proposes four common perspectives in modelling BP. From the functional perspective, a model represents what process elements are performed. These process elements may consist of objects, data, artifacts, or products. Therefore, in the behavioral perspective, a model represents when process elements are allocated (e.g. sequencing), and how related actions are performed. From an organizational perspective, a model represents where and by whom in the organization process elements are performed. However, from an informational perspective, a model represents the informational entities produced by a process, their structure and their entities; such as data, documents.

These four modelling perspectives cover the essence of BP, includes what, when,
where and by whom the process elements are performed and how related actions are executed. However, the model verification and validation and the modelling procedure should be considered after reviewing various BP modelling methods [79]. The progress of the process uses resources and can be conditioned by events of internal or external origin. The arrangement of activities corresponds to the structure of the process. The most commonly recognized process typology distinguishes three categories according to [35]:

- **The steering or management processes**: processes aim to organize the strategic objectives of the company.

- **The BP (or operational process, realization process)**: they have the function of accomplishing a naked mission in a given domain and use several functions of the company.

- **The support processes** are peripheral to the business of the company and only participate directly in the accomplishment of a business objective.

According to [126], the fourth category of the process is grafted around these categories, called *measurement process*. Processes make it possible to provide the metrics necessary for the improvement of the processes and their continuous improvement. The following section will discuss research focus.

## 2.3 Business Processes

The notion of *Process* has become an important asset for daily life in organizations and enterprises. It helps achieve organizational objectives and defines IS management. It has a relevant role related to the concept of information. The use of BP can be considered diverse according to the view of the company or the structure of resources used.

### 2.3.1 Process Notion Definition

We can define a process [54] as a sequence or series of steps, operations, or actions undertaken to achieve a particular goal and desired outcome. It is also affected by events occurring in the external world.
Another definition given by Davenport [31] seems to support Eriksson and Penker’s view when he describes a process as: ‘simply a structured set of activities designed to produce a specified output for a particular customer or market’. It implies a strong emphasis on how work is done within an organization, in contrast to a product’s focus on what is done. A process is thus a specific ordering of work activities across time and place with a beginning, an end and clearly identified inputs and outputs: a structure for action. Jacobson [61] describes processes as cutting across traditional hierarchies and for them to succeed they should involve collaboration between individuals or groups to accessed replaces to achieve a goal.

Jacobson says a customer-orientated process is expressed in terms of meeting an individual customer’s needs, by concentrating on processes that provide value to customers and not merely to its other parts. We arrive at what the organization should be doing when Taylor developed his theories of scientific management and refined the work activities to simplify, deskill, and specialize, workers, he still carried out a process to achieve a goal.

As an example of a process, we propose the travel reservation process that includes different sequence of steps of the process scenario: find a flight; flight booking; find a taxi; taxi booking; find a hotel, and booking a hotel. The figure below 2.1 illustrates the process:


Figure 2.1: Travel Reservation.
Processes are divided into two categories: abstract and executable processes [74]. These two kinds are specified at different abstraction levels. An abstract process only describes the control-flow while an executable process described the data-flow as well as other aspects related to the execution, such as services bound to activities. Thus, an abstract process is a model of its corresponding executable process. When the objective of the process relates to business the process will be called BP.

### 2.3.2 Business Process Definition

Taking into consideration various references, there are several perspectives defining and conceptualizing BP:

The definition below is combination of [20] and [83] in order to well define the concept:

**2.1. Definition.** BP is a collection of activities that takes one or more kinds of input and creates an output that is of value to the customer. Generally, it is a set of activities undertaken in a specific objective that can accomplish a specific organizational goal.

Another definition from [57] mentioned that:

**2.2. Definition.** A BP is a step by step algorithm to achieve a business objective when the steps of the process are called activities.

The Workflow Management Coalition (WfMC) defines BP as:

**2.3. Definition.** A BP is a set of procedures or activities related to each other to collectively achieve a business purpose by challenging roles and functional interactions within an organizational structure.

In addition, this definition given by [32] describes the term BP as:

**2.4. Definition.** BPs are a structured set of work activities that lead to specified business outcomes for customers.

Grover and Teng [51] introduce BP as well that:

**2.5. Definition.** A BP is nothing more than logically related tasks that use the resources of a company to achieve its defined business outcome.

The term BP is intended to embrace not only the control flow [72]; i.e. the chronological sequence of function execution but also the descriptions of data, organizations,
and resources that are directly associated with it.

The definition of BP can be simple including solely a sequence of activities, or a complex including a controlled composition of services and activities performed under condition, in parallel, or encapsulated in sub-processes.

In order to define a BP, it is necessary to define:

- Activities which are the basic elements of a BP and assignment of resources;
- Business services providing a business operation;
- Business logic, including the rules that define the control through flow during the BP execution.

A detailed consideration of the various referenced definitions is excluded from the scope of this dissertation. The focus of this dissertation is on a BP model and the definitions listed in this paragraph are all acceptable definitions to accomplish the set goals. With the definition of BP in place, it is feasible to explore a BP model concept.

BP consists of a group of business activities undertaken by one or more organizations in pursuit of some particular goal for example, booking flight tickets and B2B transactions [70]. A BP is an elementary component of the overall process in the organization which can be defined as a composite set of tasks that comprise coordinated computer-based and human activities. Different examples could be as an illustration of BP mentioned in different domains such as administration, education, medicine, economy, banking...

The most typical known example of a BP is the processing of a purchase order. The activities constituting this process begin with taking into account the order and end with delivery to the customer. Between these two extremes, several intermediate activities take place along a path known in advance to guarantee the correct and consistent operation of this process. An example of these intermediate activities is the verification of stock level for the requested items and in the case of a negative response, the manufacturing process will have to be launched [77]. In case where the manufacturing process requires a supply process, a connection to an external process

---

becomes essential which introduces the common process of the supply chain. When a process definition is an input to an engine, the engine can run an instance of the process, defined as:

2.6. Definition. *A running instance of a BP represents a real case of an operational process of a company and consists of structures of activities for the process in question.*

BP can be classified as Business Process Management Initiative \(^4\) (BPMI). According to the frequency of execution in:

- (i) *Ad-hoc process* \(^5\): executed once, e.g. building a supply chain;
- (ii) *Production process*: high frequency, e.g. the construction of a series of houses of the same type;
- (iii) *Mass production process*: very high frequency, e.g. car production.

However, in the hierarchical administrative level, BPs can be classified in:

- (i) *Primary process or production process*: ex. block production in a construction company;
- (ii) *Secondary processor support process*: ex. handling of production tools;
- (iii) *Tertiary process or management process*: ex. project management.

The classification of BPs allows us to analyze the different requirements regarding the modelling of processes according to their type. For example, when it is a production process, the focus is on building logic to minimize execution time. The construction or assembly logic is also important for the composition of the common BP. However, in the latter case, the objective will be to guarantee its proper functioning. To this end, an effective management tool for BP is necessary to facilitate the intervention of actors and users on the different process levels.

After the definition of the BP concept, it requires being managed, monitored, and supervised by BPM, the following paragraph deals with the concept of BPM

\(^4\)https://searchcio.techtarget.com/definition/Business-Process-Management-Initiative-BPMI

\(^5\)An ad-hoc workflow can be seen as a non-predefined orchestration of several activities (or sub-processes) brought together to achieve a predefined objective. it is regarding the nature of its sub-process. Indeed, the tasks in an ad-hoc Workflow are built according to the organization of the process in the real world and not according to a rigid framework.

23
and its function.

2.3.3 Business Process Management

As we mentioned, BP have an important role in the organization for that reason it should be managed well. BPM is the function of studying methods tools and techniques that support and help BP, its analysis and its design to assure and support different phases BP life-cycle.

An organization can be viewed as a collection of BP that change or grow with time in response to signals from the business environment. To allow organizations to study their processes and identify change opportunities more effectively, BPM has emerged as a set of techniques and tools for understanding and redesigning processes with greater ease and increased efficiency. BPM definitions are diverse, however, in most cases it is referred to [128] as below:

2.7. Definition. The support of BP is done by using methods, techniques, and software to design, enact, control, and analyze operational processes involving humans, organizations, applications, documents, and other sources of information.

From this point of view, BPM is considered as a management approach that promotes the perception of the organization as a system composed of interconnected BP. Traditionally, BPs are executed manually and following the business rules of the company. But currently, BP take advantage of technological advances and are supported by specific software called BPM Systems (BPMS), defined as following:

2.8. Definition. BPMS is a generic software used to coordinate the execution of the various activities of the organization. It is guided by explicit BP representations.

After the definition of BPM concept the next will be BP life-cycle.

2.3.4 Business Process Life-Cycle

Organizational environments are focusing on BP following the BPM approach, this approach aims to create a life-cycle and support of the manager during the various phases. It helps ensuring continuous improvement from the process design until the
optimization step according to the business goal and organization environment.

There is no uniform view of the number of phases of the life-cycle steps but it differs according to the granularity chosen for the identification of the phases. In sum, there are four phases: *model (design), implement, execute, monitor and optimize*. These phases are illustrated in the figure 2.2:

![Business Process Management Life-Cycle](image)

1. **Process modelling**: during this phase, the representation of the model will get as close as possible to reality "to-be" processes with the objectives of reducing problems and increasing efficiency. Business experts define it in a detailed manner using graphical modelling tools, generally adopting the Business Process Model Notation (BPMN) standard. They specify the sequence of tasks in the BP. Due to a lack of information techniques (*exception handling, data formats*) for the activities, the known model must be transformed into an executable process model.

2. **Process Implementation**: the process created in the previous phase is enriched to be executed by a BP Engine (BPE). Such a standard language BPEL
is used for the listening process and to express them in events in the BP. The executable process model obtained can be present to be supported by a process engine that will implement the written regulations.

3. **Process Execution**: the executable process is interpreted by the BPE process engine. It (BPE engine) is responsible for interactions between process elements: *(documents, information, and treatments)*. It executes process instances while assigning automatic tasks to web services and manual tasks to actors. More concretely, it is considered as the implementation of a BPM solution that is linked to the company’s IS (applications and databases).

4. **Process Pilotage and Optimization**: the available information is used to evaluate and optimize process models and their implementation. This phase consists of an analyzer and study the state of processes, check the smooth running activity through the analysis of the dashboards. Prove the performance of processes and improvement of the implemented processes.

Modelling BP consists of the following phases: initially, examining and modelling the organizational structure, next, examining and modelling the existing BP and creating a base of the company’s BP, then verifying BP, after that analyzing vulnerable points, and finally modelling advanced BP. Modelling the existing situation enables the process to become clear and available to each employee. The basic advantage of this phase is the unified look BP which frequently leads to the creation of new ideas on how to advance the processes [135]. Commonly, while documenting the processes, redundancies and illogical areas are unveiled in the existing BP, which were either not visible earlier, or had been interpreted conversely.

In that case, advancements are integrated into the existing state. Some analyses show that up to 80% of process advancements are defined during modelling the existing status phases [116]. At the end of this phase, a database of BP in the company is created. After the model of advanced BP is created, it is necessary to measure the objects of the process. Following implementation, the altered BP needs to be constantly monitored. Beginning with static controls of the process, which track its stability through many dynamic analyses including simulation investigating costs of operations to determine satisfaction levels of internal and external customers, as well as the owners of the process, and the employees and deliverers of the process.

26
To manage BP, we need to document, analyze, and model them.

IT permits us to feed the described (modeled) BP into databases adapted for the storage of BP models. Such a database is a key resource for management in making strategic decisions inside their organization. In the majority of cases, the analysis of BP models is based on hand-entered parameters such as time required to execute a given function, waiting-time, availability, and utilization of resources, etc.

In cases where the BP are supported by IS, there is a transaction base that contains data on the processes, and it is necessary to realize an interface for the BP database and to develop components with the task of exporting data from the production databases of a given IS and importing that data into the analytical bases, that is, to give parameters to the BP database. In this way, the simulated process can be altered in a step-wise fashion, supplementing the process model with the essential parameters, in the beginning, then gradually measuring and testing simulation models by improving the simulation and bringing it closer to the true level of operations.

2.4 Modelling of Business Processes

In this section, we present the notion of the BP model, its role, and a broad study of existing models to deliver a good comparison between them based on various criteria.

2.4.1 BP Model Definition

Within a definition for a BP in a place, the next step is to define the BP model, which is generally an abstract representation of reality that excludes much of the world’s infinite detail. BP modelling tools [92] allow process representation is lighter digital way that can then be transferred to a live automated process.

According to [110] BP model is defined as the following:

2.9. Definition. *BP model is the abstraction of how systems and individuals collaborate to meet a business need described in notation and representation of knowledge and expertise of the practice of the profession.*

Another definition refers to [138] as below:
2.10. Definition. The model is an abstract representation of reality. Details that are unnecessary are not included as a rule in most modelling efforts. The modeler determined which aspects of the real system are of interest and which system elements are to be modeled.

A BP model or schema is a formal representation of work procedures that controls the sequence of performed tasks and the allocation of resources. Ultimately BP model must support multiple levels of obstruction serving abstraction [28] to serve various needs. The BP model must present all the elements as well as the objective or the reason behind the creation of the process, the event causing its initialization, the inputs, the outputs, the resources consumed, the activities carried out, and the execution sequence. For generalization reasons, BP can be understood as a recursive grouping of sub-processes until we reach the level of tasks that are non-decomposed activities.

2.4.2 Usefulness of Business Process Models

The purpose of a model is to reduce the complexity of understanding or interacting with the phenomenon by dropping the details that do not influence its relevant behavior. A BP model is an abstract description of an actual or a proposed BP representing selected BP elements. These elements are considered important to the purpose of the model and can be enacted by humans or machines.

Various BP modelling tools allow us to represent the process in a digital way that can then be transferred to a live automated process. Its benefits are:

- A BP model is a kind of plan that guides the flow of the trade from the start to the end to give a clear understanding of how the process works;
- Provides the basis for improving the process;
- Serves as a basis for decision support, effects decisions by setting priorities on objectives, work as a basis for obtaining resources;
- Provides consistency and controls the process;
- Identifies and eliminates redundancies and inefficiencies;
- Allows us to anticipate changes and evolution. Even if it does not provide all
the answers, it affords a vision of the strategy to be followed by the company;

- Automate BP guidance requires automated tools for manipulating BP descriptions;
- Support BPM requirements.

### 2.4.3 Business Processes Representation (Models)

The modelling of BP makes it possible to represent the functioning of a process and defines all the activities to be carried out as well as their execution orders. According to [28] a modelling technique should be capable of representing one or more of the following modelling perspectives: functional (represents what activities are being performed), behavioral (represents when and how activities are performed), organizational (represents where and by whom activities are performed) and informational (represents the informational entities-data). BP change involves changes in people, processes, and technology over time. Most of the currently used enterprise modelling technologies can be considered as static. In real-life scenarios, BP is not static. That’s why the demand for non-static models appears. The references [103] and [14] contain a brief comparison between static and dynamic modelling:

**Dynamic** model helps the display of activities (*the steps of the process are called activities*) and flow of events within a process. The advantage of using a dynamic aspect in modelling is that it enables the outcome of a change process to be evaluated before it is implemented into the physical environment. Where *Static* models have deterministic nature and are independent of the process sequence, it may depend on the data collection and documents that are processed during the flow of information.

Currently, most workflow languages support the basic constructs of sequence, iteration, splits, and joins. However, the interpretation of even these basic constructs is not uniform and it is often unclear how more complex requirements could be supported. Indeed, vendors are allowed to recommend implementation levels such as database triggers and application event handling. The result is that neither the current capabilities of workflow languages nor insight into more complex requirements of BP is advanced. Process models are only useful if they help to improve processes. For instance, verifying the absence of deadlocks in models is a prerequisite for process automation. However, models that are sound but at the same time not used to
CHAPTER 2. LITERATURE REVIEW

configure a BPM (BPM is the study the design and the execution of the process [57]) system do not improve performance. BP needs to be managed in environments where processes are only partly documented, and a range of IS is used. These systems are often “unaware” of the processes in which they are used [42].

2.5 Business Process Technologies

In this section, we describe some technologies that participate in the implementation and management of BP.

2.5.1 The Concept of Workflow

A process should not be confused with a workflow. Indeed, unlike BP which represents a set of activities and procedures that collectively allow the achievement of a business objective, the workflow describes the partial or entire automation of a BP [146]. It is used to provide each actor of a coat with the necessary information for the execution of activities that combine it, namely documents, information, and tasks, according to a set of procedural rules WfMC. In other words, it is a particular representation for which the coordination mechanisms between activities, applications, or participants can be managed by a workflow management system (WFMS).

The concept of workflow appeared in the early 90s in the context of research on software tools facilitating collaborative work. WfMC [65][76] defines workflow as:

2.11. Definition. The automation of a BP, in whole or part, during which documents, information, or tasks are passed from one participant to another for action, according to a set of procedural rules.

Similarly to the workflow, the WFMS is defined as:

2.12. Definition. A system that defines, creates, and manages the execution of workflows through the use of software running on one or more workflow engines. It can interpret the process definition, interact with workflow participants and, where required, invoke the use of IT tools and applications.

A workflow, thus, automates the flow of information between different processes within the company. The workflow is composed of activities, corresponding to tasks or functions to be carried out to complete a stage in the process. A process must be
modeled before being implemented in a workflow engine.

### 2.5.2 Workflow Management Systems

A WfMS is a system that defines workflow processes, creates and manages the execution of these instances. This execution is controlled by a workflow engine that can interpret the definition of the process, interact with the different participants, and start the execution of one or more programs or applications.

A model workflow process is made up of a set of logical steps, called activities, automatic or interactive, as well as a set of relationships (sequence, parallelism, synchronization, etc.) allowing them to be linked, a set of criteria to trigger or interrupt a process, of a set of information, such as the actors that can be a human or an automatic resource and who must execute an activity, the data, or the associated computer applications.

The WfMS allows regrouping principles, methodologies, and technologies issues from several IT fields and management sciences. BPs are modular. These modules can be reorganized by the Workflow Manager to build a new BP; to react rapidly to unforeseen changes in requirements and business conditions. Besides, the Workflow Manager can track operations in real-time and integrate applications from different platforms into a single BP.

### 2.5.3 The Role of Workflow Engine

The workflow engine represents a set of tools demonstrating the ability to describe, model, automate, structure, monitor, analyze, and computerize the processes of a company. Its role [17] is summarized as below:

- Assign process players and activities to the complex business rules defined by the company;
- Provide warranty, reliability, data security, and allows a soft integration with the already existing systems;
- Help opening up interaction between business and IT experts;
- Allow graphic modelling of BP and give the ability to modify it in real-time.
The whole must give as result:

- Increased control and monitoring of BP;
- Facilitated evaluation of the efficiency of the processes;
- Anticipation of blockages and malfunctions;
- Direct feedback from users or workflow stakeholders to optimize and sequentially improve the efficiency of BP.

The essential challenge of Workflow as a BP modelling tool is to provide answers to the following questions: what are the activities (sub-processes) to be executed, in what order will the execution take place, and what information is exchanged between these activities. To meet these requirements, the workflow uses different types of association, to connect the various sub-processes in the BP: Sequence, Exclusive choice, parallel, multiple-choice, multiple merge and cycle, all those associations will be revealed in section 3.1.

2.5.4 Enterprise Architecture (EA)

The EA domain, part of the information management discipline, is about bridging the gap between business and information management. EA according to Whitman, Ramachandran, and Ketkar [138], provides the mechanism by which the reality of the enterprise and its systems can be aligned with management intentions. The business architecture, one of the EA dimensions, includes BP models, reflecting how activities are coordinated in the course of a BP.

EA has not been well defined and agreed upon, it’s being developed to support IS development and enterprise re-engineering. There is a widely used definition of architecture for software-intensive systems in the ANSI/IEEE 7:

2.13. Definition. The fundamental organization of a system, embodied in its components, their relationships to each other and the environment, and the principles governing its design and evolution.

An EA, therefore, shows the primary components of an enterprise and how these components interact with [18] or relate to each other to review, maintain, and control

\(^{7}\text{Standard 1471-2000 (IEEE Computer Society, 2000)}\)
CHAPTER 2. LITERATURE REVIEW

the whole operation of an enterprise. This organizing logic acts as an integrating force between business planning, business operations, and enabling technological infrastructure.

EA is the capability to envision, plan, design, lead, manage, and control organizations, systems, and/or processes in current, transitionally, and future states, and the relationships between them. It describes an organization in terms of its strategy, structure, information flows, value streams, as well as its business and transaction models. It helps businesses to understand their current composition, utility, costs, and sources of value generation. For that, we are interested in EA in the dissertation.

Enterprise architecture management (EAM) is a management philosophy concerned with corporate change. It helps the improving of an enterprise’s performance and results in clear architecture principles and guidelines. These factors contribute to better alignment and reduced complexity.

Why Enterprise Architecture?

The goal of EA is to design a unified IT environment (standardized hardware and software systems) across the firm or all of the firm’s business units, with tight symbiotic links to the business side of the organization and its strategy. More specifically, the goals are to promote alignment, standardization, reuse of existing IT assets, and the sharing of common methods for project management and software development across the organization. As a result (from a theoretical point of view), the EA will make IT cheaper, more strategic, and more responsive.

The purpose of EA is to create a map of IT assets and BP and a set of governance principles that drive an ongoing discussion about business strategy and how it can be expressed through IT. Similarly, it is a governance mechanism based on business need and value to the adoption of new applications technology. Additionally, EA is improving the risk management of the organization transformation and gives enterprise-level communication. EA drives to standardization, which facilitates early design decisions, and creates a transferable abstraction of a system/environment description.
CHAPTER 2. LITERATURE REVIEW

Generic EA Components

The generic EA components described in the figure 2.3 below:

![Figure 2.3: Enterprise Architecture Components](image)

An enterprise is viewed as a complex system with a fixed boundary and an assemblage of differentiated but interdependent components. The enterprise as a whole move through various cycles is surrounded by an external environment from which it gets various inputs and into which it provides outputs. Inputs are transformed by enterprise components to produce outputs in the form of products and services that are returned to the external environment.

Components are elements of the enterprise that play a part in transforming inputs and creating outputs. Elements that do not actively participate in the formulation of products or services, such as budgets and finances, are considered common supporting resources only, not components. The different components [102] mentioned in the figure 2.3 are:

- **External environment**: these are generally the several factors that are outside the enterprise boundaries but still have an impact on it.

- **Enterprise strategy**: the collection of decisions about what the enterprise will accomplish and how resources will be utilized to accomplish its desired results.

- **Corporate culture**: is the set of written and unwritten norms, values, man-
management styles, and patterns of behavior that affect enterprise functions.

- **People**: human resources of the enterprise with their attitudes, behavior, and their readiness to accomplish the enterprise mission.

- **Organizational structure**: Are the different organizational arrangements (formal and informal) created to accomplish the overall mission.

- **Technology**: the various equipment, machinery, methods, and tools important to accomplish the tasks; it includes hardware, software, and telecommunications.

- **Information**: this is the knowledge and data created/used by people, processes, and technology.

- **Processes**: are the flow of activities that enable the enterprise to carry out its work and produce its products and services.

- **Tasks**: are the work performed or activities carried out by the enterprise and its parts.

- **Enterprise results**: are the products and services produced by the enterprise for the customer.

**Enterprise Architecture Frameworks**

To cope with architecture complexity, different frameworks, methods, and tools have been developed. An EA framework is a set of assumptions, concepts, values, and practices that constitutes a way of looking at enterprise reality via views on (architectural) models. It offers a fundamental structure, serving as a scaffold for developing, maintaining, and using EA. It could be also described as a collection of descriptions and methods to create and manage EA.

Several EA frameworks are available, e.g. the IEEE Standard 1471-2000 \(^8\), the Zachman Framework \([144]\), the The Open Group Architecture Framework (TOGAF) \([11]\) or the DODAF framework\(^9\). The most recognised frameworks are the Zachman Framework (for rather theoretical purposes) and the TOGAF framework (for rather practical usage). The development of EA frameworks started in the 1980s. The table

---

\(^8\)Software architecture: Introducing IEEE standard 1471. [80]

2.5.4 below collects the different suggested EA Frameworks (Zachman, NIST\textsuperscript{10}, TAFIM\textsuperscript{11}, Spewak, TOGAF, C4ISR\textsuperscript{12}, TISAF\textsuperscript{13}, FEAF\textsuperscript{14}, TEAF\textsuperscript{15}, FEA\textsuperscript{16}, DoDAF\textsuperscript{17}, E2AF\textsuperscript{18}) and their types:

<table>
<thead>
<tr>
<th>Release Date</th>
<th>EA Framework</th>
<th>Source</th>
</tr>
</thead>
<tbody>
<tr>
<td>1987</td>
<td>Zachman Framework</td>
<td>Commercial Framework</td>
</tr>
<tr>
<td>1989</td>
<td>NIST Enterprise Architec-</td>
<td>Government Framework</td>
</tr>
<tr>
<td>1991</td>
<td>TAFIM framework</td>
<td>Defence Industry Framework</td>
</tr>
<tr>
<td>1992</td>
<td>Spewak’s EA Planning</td>
<td>Other Framework</td>
</tr>
<tr>
<td>1996</td>
<td>TOGAF framework</td>
<td>Enterprise developed Frame-</td>
</tr>
<tr>
<td>1996</td>
<td>C4ISR framework</td>
<td>Defence Industry Framework</td>
</tr>
<tr>
<td>1997</td>
<td>TISAF framework</td>
<td>Government Framework</td>
</tr>
<tr>
<td>1999</td>
<td>FEAF framework</td>
<td>Government Framework</td>
</tr>
<tr>
<td>2000</td>
<td>TEAF framework</td>
<td>Government Framework</td>
</tr>
<tr>
<td>2002</td>
<td>FEA framework</td>
<td>Other Framework</td>
</tr>
<tr>
<td>2003</td>
<td>DoDAF framework</td>
<td>Other Framework</td>
</tr>
<tr>
<td>2003</td>
<td>E2AF framework</td>
<td>Defence Industry Framework</td>
</tr>
</tbody>
</table>

Table 2.1: The Most Established EA Frameworks (Development and types).

Using an architecture framework will speed up and simplify architecture development, ensure more complete coverage of the designed solution, and make certain that the architecture selected allows for future growth in response to the needs of the business.

In fact, the most knowledgeable and used frameworks are the *Zachman Framework* and the *TOGAF Framework*. Firstly, Zachman architecture developed for IS in the

\textsuperscript{10}NIST: National Institute of Standards and Technology
\textsuperscript{11}TAFIM: Technical Architecture Framework for Information Management
\textsuperscript{12}C4ISR: Command, Control, Communication, Computers, Intelligence, Surveillance, Reconnaissance
\textsuperscript{13}TISAF: Treasury Information System Architecture Framework
\textsuperscript{14}FEAF: Federal Enterprise Architecture Framework
\textsuperscript{15}TEAF: Treasury Enterprise Architecture Framework
\textsuperscript{16}FEA: Federal Enterprise Framework
\textsuperscript{17}DoDAF: The Department of Defense Architecture Framework
\textsuperscript{18}E2AF: Extended Enterprise Architecture Framework
enterprise. John Zachman based his framework on practices in traditional architecture and engineering. This framework represents the whole enterprise in descriptive building blocks. Its main advantages are methodologies Independence and smooth understandability.

Figure 2.4 depicts the framework pictorially. The *vertical axis* (The columns are called *Abstractions*) provide multiple perspectives of the overall architecture. They answer key questions about the enterprise such as *What* (data), *How* (function), *Where* (network), *Who* (people), *When* (time), and *Why* (motivation). However, the horizontal axis provide a classification of the various artifacts of the architecture.

The *rows* are called Perspectives. Each row represents the viewpoint of a single group of stakeholders. Similar to other frameworks, its purpose is to provide a basic structure that supports the organization, access, integration, interpretation, development, management, and transformation of a set of architectural representations of the organization’s IS; such objects or descriptions of architectural representations are usually referred to as artifacts:

![Zachman Framework Diagram](image)

**Figure 2.4: Zachman Framework**

In 1987, John Zachman [85] wrote: “To keep the business from disintegrating, the concept of IS architecture is becoming less of an option and more of a necessity”. From that assertion over 20 years ago, the Zachman Framework has evolved and become the model through which major organizations view and communicate their enterprise information infrastructure.

Secondly, TOGAF is developed by Open Group for software systems within com-
panies. It is a method that contains two main parts: The Architecture Development Method (ADM) and the Foundation Architecture with generic functions/services on which specific architectures and building blocks can be built. TOGAF is a process-oriented EA framework that breaks an EA into different EA layers. It consists of 10 phases, Phase B-D cover the four architecture domains (1-4), respectively.

Architecture domains are considered different conceptualizations of an enterprise. TOGAF provides four architecture domains: business, data, application, and technology architectures. The figure below 2.5 shows the main structure of the architecture domains according to the TOGAF framework:

![Figure 2.5: TOGAF Architecture Domain Structure.](image)

TOGAF plays an important role in helping to systematize the architecture development process, enabling IT users to build open systems-based solutions to their business needs. Of course. These frameworks are certainly not the only ones that constitute the EA body of knowledge, as pointed out earlier. We do not intend to endorse these two frameworks as the best ones. However, these two are the most widely known approaches today. Due should be followed by a negative thing this could be explained because of their versatility, visibility, and market impact so far.

### 2.5.5 Web Services

For years, designers and developers of IS components have tried to answer the question: how to dynamically modify, increment, and link heterogeneous applications to meet business requirements. The web services paradigm has emerged as a powerful mechanism for integrating distributed IS. Combining the best aspects of development in the field of building component-based IS with the facilities offered by the web.

In this section, we will introduce some basic concepts and definitions of Web services, as well as their characteristics and life-cycle to specify the relation between
Web services Definition and Characteristics

Web services have emerged over the past decade as a powerful solution for integrating distributed and heterogeneous IS, this concept is defined as:

**2.14. Definition.** Web services are open standard (XML\(^{19}\), SOAP\(^{20}\), HTTP\(^{21}\), etc.) based Web applications that interact with other web applications to exchange data. Web services can convert existing applications into Web-applications. It is an application (accessible function from the network) that can be consulted and invoked using standard Web protocols.

The World Wide Web Consortium (W3C)\(^{22}\) defines a web service as an application or software component that checks the following properties:

- It is identified by a Uniform Resource Identifier (URI);
- Its interfaces and links (binding) can be described in XML;
- Its definition can be discovered by other Web services;
- It can interact directly with other Web services across the XML language and using Internet protocols.

The goal of Web services

The main goal of Web services is to achieve universal interoperability between applications by using Web standards. They use a loosely coupled integration model to allow flexible integration of heterogeneous systems in a variety of domains including business-to-consumer, business-to-business, and enterprise application integration. The following basic specifications originally defined the Web services space: SOAP, Web Services Description Language (WSDL), and Universal Description, Discovery, and Integration (UDDI). SOAP defines an XML messaging protocol for basic service

\(^{19}\)XML: Extensible Markup Language  
\(^{20}\)SOAP: Simple Object Access Protocol  
\(^{21}\)HTTP: Hypertext Transfer Protocol  
\(^{22}\)http://www.w3.org/2002/ws/
interoperability \(^{23}\). WSDL introduces a common grammar for describing services. UDDI provides the infrastructures required to publish and discover services in a systematic way. Together, these specifications allow applications to find each other and interact following a loosely coupled, platform-independent model.

The outcome of the Web services approach is to transform the web into a distributed computing device where programs (services) can interact intelligently by being able to automatically discover, negotiate, and compose themselves with more complex services. Following are the benefits of using web services:

- Exposing the existing function on the network;
- Web services allow us to expose the functionality of the existing code over the network;
- Interoperability, i.e Web services allow various applications to talk to each other and share data and services among themselves;
- Standardized Protocol: for the communication, all the four layers (Service Transport, XML Messaging, Service Description, and Service Discovery layers) use well-defined protocols in the web services protocol stack;
- Low-cost communication: Web services use SOAP over HTTP protocol.

Components of web services

The basic web services platform is \textit{XML+HTTP}. All the standard web services work using the following components: \textit{SOAP}, \textit{UDDI} and \textit{WSDL}.

A web service uses XML to tag and format the data. It employs SOAP to transfer a message and WSDL to describe the availability of service.

Conclusion

In this chapter, we exhibited various concepts, definitions, and terms used in our research. In the next chapter, we will focus on the BP models and do a deep investigation of them.

\(^{23}\)Web services allow various applications to talk to each other and share data and services among themselves. Web services are used to make the application platform and technology independent.
Chapter 3

Business Process Models

We often model BP to represent activities, items, entities, their relation, and how to intercommunication between them in an enterprise needed for their success. BP modelling and management today emphasize the term BP because of its great importance in an enterprise render available business and data in a format that designers can use for models.

3.1 Existing Business Process Models

BP modelling is one of the most commonly used techniques in requirements analysis, often taking data flow diagrams as a form. There are several newly developed methods for modelling and these techniques vary significantly. Most BP modelling approaches focus on some diagram-like graphics, which reveal the coordination between activities in BP.

The BP diagrams should contain activities and activities connectors, a way to represent decision points, and other strategies to express various activity coordination patterns, such as sequential flow, branching, and parallel execution. Following defined are various basic patterns BP models:

- **Sequence**: is an ordered, or things following each other, or series of activities where each activity starts after completing the previous one.

1Patterns mean the recurrence of same basic features during the life-cycle of a process
• **Exclusive choice:** could be perceived as a branch that comes from a single activity and goes to precisely one of several paths. The path is chosen either by using a decision or by the mean of a condition.

• **Parallel split:** is defined as a gateway that uses a simple control link to partition two or more activities. These activities will be processed and running in parallel. We could also define it as a branch from a single activity and get dispatched to multiple parallel paths.

• **Multiple choice:** compared to the exclusive choice, it allows the choosability between all possible alternative paths at performance time. its execution is by Choosing only one or more parallel branches that satisfy a particular condition.

• **Multiple merge:** is a location during the process where multiple paths are merging but without any control, and it is a point in which one or more branches of the control thread join without synchronization.

• **Cycle:** is a mechanism that uses some conditions to repeat a collection of patterns defining a set of activities, actions, work tasks; i.e., the same instructions multiple times.

In the literature, there exist different models and languages for BP modelling; the most used and known ones are:

### 3.1.1 Petri-nets

The classical petri-net was invented by *Carl Adam Petri* in the sixties\(^2\). Since then, researchers used it to model and analyze all kinds of processes with applications ranging from communication protocols, hardware, and embedded systems to flexible manufacturing systems and user interactions. Recently, researchers extended the classical Petri-net with color, time, and hierarchy [6]. These extensions facilitate the modelling of complex processes where data and time are important factors.

Petri-nets gained an increase in usage and acceptance as a method for describing information flow and control [96]. As a modelling language, it graphically depicts the distributed system’s structure as a directed graph with annotations. Such petri-nets have place nodes, transition nodes, and directed arcs connecting places with

\(^2\)https://www.computer.org/profiles/carl-petri
transitions. Petri-nets model the dynamic behavior of systems. The places of a petri-net contain tokens; firing of transitions removes tokens from the input places of the transition and adds them to its output places.

**Example:** The figure 3.1 shows petri-nets firing rules. (A) An unmarked net with one transition t1 with two input places, p1 and p2, and one output place, p3. (B) The marked net, the net with places populated by tokens; the net before firing the enabled transition t1. (C) The marked net after firing transition t1, two tokens from place p1 and one from place p2, are removed and transported to place p3.

![Figure 3.1: An Example of Petri-nets.](image)

### 3.1.2 Finite State Machines (FSM)

An FSM is a well-known modelling formal system specification. It is also a standard model used in the mathematical foundation of computer science. This model concept is defined and discussed in innumerable books and papers. An FSM is a behavioral model that contains states and transitions. In our dissertation, FSM representation is often used as a BP model; the focus of this section. We will present its components and its utility in detail.

FSM is composed of three sets: a set of *states*, a set of *actions*, and a set of *transitions* labeled between states, an *initial* state and *final states*. It has a 5-element tuple: \( A = (M, Q, q_0, F, R) \) where:

- \( M \): an input alphabet;
- \( Q \): finite set of states of \( A \);

\(^3\text{https://www.sciencedirect.com/topics/computer-science/petrinets}\)
CHAPTER 3. BUSINESS PROCESS MODELS

- $q_0 \in Q$: is the initial state;
- $F \subseteq Q$: is final states;
- $R : Q * X \longrightarrow Q$: is the transition function.

Figure 3.2 shows the structure and the different elements of the FSM where $s_0$ is the initial state, $s_f$ the final state, and $a, b, c, d, e, f, g, i$ are transitions.

Figure 3.2: An Example of FSM structure.

Basic FSM Patterns

Various FSM patterns are defined as the basic modelling patterns of BP listed before 3.1.2:

- **Sequence**: is an ordered series of activities. Each new activity starts after the previous one finishes. Figure 3.3 shows the structure of a sequence of four states (Where: $s_0$: initial state, $s_3$ the final state, and $T_1; T_2; T_3$: the set of transitions between states) in FSM.
CHAPTER 3. BUSINESS PROCESS MODELS

Figure 3.3: Example of Sequence Structure in FSM.

- **Multiple choice**: or switch differs from exclusive choice pattern in that the multiple-choice pattern allows at least one alternative paths to be chosen at performance time. Technically, a Multiple-Choice pattern may allow zero paths chosen, but this could be considered an invalid situation where the process flow stops unexpectedly. A multiple-choice example shown in Figure 3.4 (Where $s_2$ or $s_3$ executed only if $T_1$ or $T_2$ true) below:

Figure 3.4: Example of Multiple Choice Structure in FSM.

- **Parallel split**: happens when two or more activities start at the same time. It is a point in the process where we split a simple control link into multiple control links running in parallel. Parallelism, when finishing the execution of the main activity, the control link goes to activities sub-activities or sub-states the same time as the example below 3.5 (the states: $s_1; s_2$ and $s_3$ executed in
parallel) shows.

![Figure 3.5: Example of Parallel Split Structure in FSM.](image)

- **Multiple merge:** is a location in the process where multiple paths merge without any control. We also consider it a point in which one or more branches of the control thread join without the need for synchronization. Generally, the example in Figure 3.6 below.

![Figure 3.6: Example of Multiple Merge Structure in FSM.](image)

- **Cycle:** is a mechanism that repeats the same instructions multiple times with conditions. Generally, we execute from top to bottom and, once at the end of the loop, it starts again at the first instruction, then it begins again and repeats the same way as Figure 3.7 shows below.
CHAPTER 3. BUSINESS PROCESS MODELS

3.1.3 Business Process Model Notation (BPMN)

BPMN is a fundamental ontology that represents domain information model entities. BPMN [118] creates a standard to fill the gap between the design and implementation of BP. It provides a visual language in the form of graphical notation for defining BP in a diagram.

BPMN is a standard language for describing BP, particularly at the domain analysis level and high-level system design. A growing number of process design, EA, and workflow automation tools provide modelling environments for BPMN.

The most significant advantage of BPMN is that it’s a standard with well-defined syntax. So many business analysts are familiar with it, which makes collaboration much more effortless. Also, most modelling tools support BPMN 4, which makes it much easier to share and edit even if using different softwares. All these make BPMN the most popular BP modelling technique at the moment.

Figure 3.8 describes an example where an order of iPhone from eBay 5 is expressed by a BP:

This standard provides organizations with the capability of understanding the internal business procedures in a graphical notation. Also, it gives the ability to

4https://creately.com/blog/diagrams/business-process-modelling-techniques/
5https://creately.com/blog/examples/bpmn-templates-model-processes/
CHAPTER 3. BUSINESS PROCESS MODELS

Figure 3.8: BPMN Diagram of iPhone Ordering Process from eBay.

communicate these procedures in a standard manner. Furthermore, the graphical notation facilitates the understanding of the performance collaborations and business transactions between organizations.

3.1.4 Unified Modelling Language (UML)

UML is one of the most used modelling languages. UML is a meta-model with several packages; each package introduces concepts expressed through graphical notation and diagrams. The UML language is a standard offered by the Object Management Group (OMG) that allows defining functional and technical needs in an object-oriented (OO) development environment using several diagrams and several concepts to increase the semantics of these models.

UML is a standard offered by the OMG, allowing functional and technical needs in an OO development environment. It uses several diagrams and several concepts to increase the semantics of these models. UML is used when modelling BP using OO analysis tools, Jacobson and Booch Rumbaugh developed UML as a universal notation for OO analysis. UML [125] offers specialized diagrams (including diagrams of activity diagram, sequence diagram, class diagram, state charts etc.) each having a specific function.

The diagram commonly used in UML to model processes is the activity diagram. In figure 3.9 [6] illustrates an example of document management process activity.

It allows the description of the process behavior in the form of flow or activity

flow. We can use UML to model BP, model the logic of the use cases or user scenarios, or model a business participant with the related business activities and business logic. The UML Activity Diagrams are the OO equivalent to the data flow, and flowchart diagrams using structured development methods.

### 3.1.5 Business Process Execution Language (BPEL)

BPEL [62][17] is a standardized language for specifying the behavior of a BP based on interactions between a process and its service partners. It defines how multiple services interact with Web Service using WSDL documents through a set of operations and messages to deal with it.

A BPEL process uses a set of variables to represent the messages exchanged between partners (they also represent the state of BP). WSDL document is through a set of operations and statements to deal with it. A BPEL process uses a set of variables to represent the messages exchanged between partners. They also represent the state of BP. A simple BP example described with BPEL is showed below:

**Example:**

```xml
<process name="Supplier process">
  <partnerLinks>
    <partnerLink name="client" ... />
    <partnerLink name="warehouse" ... />
  </partnerLinks>
  <variables>
    <variable name="BdC" type="xsd : Command"/>
    <variable name="resBDC" messageType= "string" />
    <variable name="availability" type="string"/>
  </variables>
  <sequence>
    <receive name="receiveBdC"
      partnerLink="client" portType="ptCommand"
      operation="opCommand" variable="BdC"
    >
```
In this examples 3.1.5 of BPEL code named: Supplier Process, we have two main tags: variables, partnerlinks and the part of the activities between the two tags of start and end of the process (the root element of the BPEL).

- **Partnerlinks**: there are two partners who represent the participants of the customer and warehouse process.

- **Variables**: BdC, resBDC and availability.

- **The activities part**: a sequence of activities (receive, invoke, switch) are includes between the two kicks at the start and end of the sequence.

BPEL with the XML semantics became the rapid method for information extension. We use it as an interactive language for web services. BPEL is XML-based in textual format and contains complex constraints. The article [117] used to model BP and provides fundamental ontology to represent domain information model entities.

### 3.1.6 Process Algebras

In computer science, the process algebras (or process calculi) is a diverse family of related approaches for formally modelling concurrent systems. Process algebras provide a tool for the high-level description of interactions, communications, and synchronizations between a collection of independent agents or processes [7].

Process algebra is specification approach of communication between processes
CHAPTER 3. BUSINESS PROCESS MODELS

and events. Its essential components are the syntax as determined by the well-formed combination of operators and more elementary terms. The syntax of a process algebra [89] is the set of rules that define the combinations of symbols. These symbols are correctly structured programs in that language, process algebra described by structural operational semantic (SOS) approach the various approaches are the calculus of communicating systems (CCS), communicating sequential processes (CSP) and algebra of communicating processes (ACP). The semantic-based on algebraic laws are the fundamental axioms of an equations system and process.

3.1.7 Event Driven Process Chain (EPC)

A flowchart called “event-driven process chain” (also known as merely “EPC”) is used for modelling the BP, configuring a so-called “enterprise resource planning” implementation as well as for improving the BP. It is also a modelling language invented by Prof. Wilhelmi and Nuttgens in 1992, EPC; instead a semi-formal modelling language[84] for the description of BP used for its visualization and analysis in the realm of BPM. The EPC representation allows signifying other business IS features as data structure resources and functions using nine elements, which are events, functions, a unit of organization, paths, control flows, logical connectors, information flows and materials, moreover and organizational unit assignment.

EPC is used to describe the operational sequence of processes using rules for EPC modelling. A simple event-driven process chain (Figure 3.10)\(^7\) may look as follows:

![Example of EPC](https://www.ariscommunity.com/event-driven-process-chain)

Figure 3.10: Example of EPC.

\(^7\)https://www.ariscommunity.com/event-driven-process-chain
3.2 Comparison of Business Process Models

Many researchers compared various existing models, in [134], they made a comparison between languages along two general dimensions: firstly, the contents or coverage of the language (which aspects of a BP are covered). Secondly, the intended use of the resulting process descriptions. This comparison is based on four views of a BP: informational, functional, dynamic, and organizational.

In our study, [89] we compared between several BP tools to discern the similarities and the differences based on three main viewpoints (criteria) described below:

1. **Semantic**: taking into account the meaning of the model, element labels, and comments.

2. **Syntactic**: taking into account modelling language types and language elements.

3. **Structural**: taking into account the types of modelling languages and language elements.

The table below 3.1 summarizes the comparison done:

<table>
<thead>
<tr>
<th>Models</th>
<th>Criteria</th>
</tr>
</thead>
<tbody>
<tr>
<td>Petri-nets</td>
<td><strong>Semantic</strong> &lt;br&gt;• <em>Formal semantics</em>: in classical petri-net and several enhancements (color, time, hierarchy). &lt;br&gt;• <em>Operational semantics</em>: described in terms of tokens in places in of petri-net [100]. Petri-nets have an exact mathematical definition of their execution semantics, with a well-developed mathematical theory for process analysis [146].</td>
</tr>
</tbody>
</table>
| **FSM** | FSM is a restricted Turing machine [50] where the head can only perform "read" operations, and always moves from left to right. | FSM representation is 5-let $A = (M, Q, q_0, F, R)$ where:  
• $M$: an input alphabet;  
• $Q$: finite set of states of $A$;  
• $q_0 \in Q$ is the initial state;  
• $F \subseteq Q$ is final states;  
• $R : Q \times X \rightarrow Q$ is the transition function. | Graphical nature of FSM supports communication because it is easy to understand by users and describes reactive behavior and dynamic systems. |
| **BPMN** | Semantic analysis of BPMN models is hindered by the heterogeneity of its constructs and the lack of an unambiguous definition of the notation. Formal semantics of BPMN is that of Wong and Gibbons, who uses CSP as the target formal model. | Syntactic rules are comprehensively documented in tables throughout the BPMN standard specification, the actual semantics is only described in a narrative form. | Initially positioned as a modelling formalism and only informally defined, it has matured into a fully-fledged BP modelling and execution language based on a comprehensive metamodel together with an associated graphical modelling notation and an execution semantics defining how BPMN processes should be enacted [105]. |
| **UML** | The UML semantics [105] is described in an informal manner. | UML have an abstract syntax textual notations. | UML activity models is expressed in process modelling visual language, and allows to model enterprise from different perspectives. The UML class diagrams characterize the abstract syntax of the language. |
| **BPEL** | A BPEL process is described in terms of XML. | BPEL uses a syntax based on XML (XML-based) [62] depends to the activities of BPEL. | BPEL’s activities executed in order (workflow):  
• Basic activities: invoke, receive, reply, assign, throw, wait, empty;  
• Structured activities: Sequence, Switch, While, Pick, Flow, Repeat-Until For-Each, If-Else. |
CHAPTER 3. BUSINESS PROCESS MODELS

Process Algebra

- Operational semantics: describes systems evolution in terms of labelled transitions. It is relatively close to an abstract machine based view of computation and might be considered as a mathematical formalization of some implementation strategy;
- A denotation semantics: maps a language to some abstract [34] model;
- Algebraic semantics: algebraic laws are the basic axioms of an equation system, and process.

The basic component of process algebra is its syntax as determined by the well-formed combination of operators and more elementary terms. The syntax of a process algebra is the set of rules that define the combinations of symbols that are considered to be correctly structured programs in that language.

The different operators used in process algebras will be described by relying on the so called SOS approach. The various approaches are CCS, CSP, and ACP.

EPC

- Research regarding the semantics of the EPC so far mainly concentrated on the formal semantics of the available language constructs;
- A substantial part of the semantics of an EPC model is bound to natural language.

An N-EPC is a tuple $M=(E,F,C,A,l,T,V,O,S)$ satisfying:

- $E,F,C$ are disjoint sets;
- $E$ is a set of events;
- $F$ is a set of functions;
- $C$ is a set of connectors;
- $A$ is a set of arcs, $A \subseteq (E \cup F \cup C)^* (E \cup F \cup C)$;
- $T$ is a set of terms;
- $V$ is a set of verbs;
- $O$ is a set of objects;
- $S$ is a set of states;
- $l$ is the labeling function: $l : (E \cup F \cup C \cup V \cup O \cup S) \rightarrow (V \cup O \cup S \cup T^*)$.

EPC models essentially consist of a set of functions and events, which are connected via a control flow using arcs and connectors.

Table 3.1: Comparative Analysis Of BP Models.

### 3.3 BP Models Advantages and Disadvantages

From several references and deep research about BP models, we conclude the table below 3.2, that shows various advantages and disadvantages of each model described last section:
## CHAPTER 3. BUSINESS PROCESS MODELS

<table>
<thead>
<tr>
<th>Models</th>
<th>Advantages</th>
</tr>
</thead>
</table>
| Petri-nets |  ● The plenty of petri-net verification mechanism on business models and the ability to transform many models to petri-nets;  
|           |  ● Petri-nets are minimal and very general, but also very rich in mathematical properties.                                                                                                               |
|           |  ● In system modelling of events in which it is possible for some events to occur concurrently, there are constraints on the concurrence, precedence, or frequency of these occurrences;  
|           |  ● Inability to test for specific marking in an unbounded place and to take action on the outcome of the test;  
|           |  ● Petri-nets lack means to model the timing of actions represented by transition firings, but extensions have been developed to remedy this limitation.                                                   |
| FSM      |  ● A power theory and formal methods at requirement specification, modelling, design and test case generation;  
|           |  ● Its theory has proved useful in capturing static as well as dynamic behavior of systems.                                                                                                                  |
|           |  ● Need more static memory to store the lookup table that stores the FSM events;  
|           |  ● The simplicity of FSM also became one of their disadvantages. Systems which need an indeterminate amount of states cannot be modeled by a FSM evidently.                                                     |
| BPMN     |  ● BPMN started as a purely graphical BP notation;  
|           |  ● Totally acceptable model;  
|           |  ● Existence of relation between real BP and its execution;  
|           |  ● Easy to understand.                                                                                                                                                                                  |
|           |  ● No state transitions;  
|           |  ● Hierarchical maintenance does not exist;  
|           |  ● Limited data and tasks details.                                                                                                                                                                         |
| UML      |  ● Capture activities of difficult software;  
|           |  ● Better illustration of sequence between activities;  
|           |  ● Chronological system flow;  
|           |  ● Several diagrams.                                                                                                                                                                                     |
|           |  ● UML diagrams insufficient to define semantic functionality of the system to be developed;  
|           |  ● Activity diagrams do not give much detail about behavior or interaction of objects;  
|           |  ● Do not have much mathematical foundation to represent pre/post conditions and data constraints to be useful in validation and verification of the procedures;  
|           |  ● UML has still no structure and specification for modelling user interfaces.                                                                                                                            |
CHAPTER 3. BUSINESS PROCESS MODELS

### BPEL
- Express BP using standard language;
- Describe the logic of BP through the composition of web services;
- Allow to cancel the processing done by an activity in case of failure;
- BPEL processes have a well defined life-span and their life-cycle can be controlled in order to maintain the execution of activities for a long period;
- Easy to understand because of the XML based.

### Process Algebra
- Support a variety of algebraic laws to manipulate systems and support to the whole process development both at the design stage and for reverse engineering issues;
- Process algebra is efficient at prove the conform of Web services [17] to their requirements and respect properties and it can be helpful to tackle choreography problems;
- Explain the semantics of conceptual BP models.

### EPC
- Understandable notation;
- Integration of system features;
- Representation and explanation of BP;
- Used to facilitate the adoption and customization of process oriented IS, thereby serving as a starting point for the actual implementation.

- Limited data and tasks details;
- BPEL enables the top-down realization of Service Oriented Architecture (SOA) through composition, orchestration, and coordination.

- Enable on compose larger from smaller ones in a structured way;
- Global states and global activities are not basic notions.

- An ambiguity concerning the modelling of start and end events occur in the EPC.

| Table 3.2: BP Models Advantages and Disadvantages. |

#### 3.4 Models Comparison

There are numerous BP models in the literature, we have chosen some particular of BP models that proved to be useful. To see the similarities and the discrepancies, we may highlight various viewpoints in our comparison. Mainly, we can group them based on several aspects: context, ambiguity, structure, semantic, notation, terminology and table 3.2 and table 3.3:

- Petri-net has a simple structure, and it is easy to analyze, simple and suitable for testing the model; But the low-level net is not suitable for performance analysis. To enable this function, we need to use time and color;
• Petri-net is a graphical and mathematical tool, usually applied in the simulation of discrete-event dynamic systems;

• FSM is easy to understand by users and its well known formal model that could be easily checked;

• UML Activity diagrams and BPMN are quite similar technologies, and they are suitable for static modelling of BP. Nonetheless BPMN is more convenient and has much more symbolic power as UML since BPMN covers simulation and executing process models by automating (wherever possible) the process steps;

• UML is a visual language for OO modelling approaches, it is mainly used in software modelling and helps model class connections;

• Both FSM and UML are techniques used during both design and specification phases of development depending on mathematical and computational tools;

• Petri-nets and process algebra are formal graphical representations which are understandable, because of their design in the form of a graph, they have useful connections both to graph and linear algebra which can be exploited for the verification of systems;

• Often, we use BPMN and BPEL (Initially and by design) in conjunction: BPMN for the business user-centered perspective and BPEL for the technical specification;

• Both the EPC and UML activity diagrams have similar concepts and have different contexts under which they are developed. They focus on modelling a system (process inside the system). No restrictions are appearing to exist on the structures of EPC, but non-trivial structures can involve parallelism and have ill-defined execution semantics. Once they are as we described, they resemble UML activity diagrams;

• The UML models are richer of concept than others because of the higher view of modelling for specific cases, but EPC structure can be expressed more efficiently rather than UML.
CHAPTER 3. BUSINESS PROCESS MODELS

3.5 Model Transformations

Nowadays, the variety and complexity of applications arise the requirement for creating new flexible models. The transformation between models becomes more significant.

Various research studies have dealt with the problem of transformation in general [38]. These approaches vary from the conversion between the different models or models to language or language to model. The model transformation is seen as the heart and soul of Model-Driven Development (MDD) [114], and it is frequently applied in many research domains.

Transformations summarized in 3.11, are from model-to-model, text-to-model, or model-to-text by defining the mapping rules to transform elements from their meta-models.

Figure 3.11: Model Transformation.

In this section, we present many transformations. First, the paper [91] aims to review main current techniques in translation from graph-oriented workflow languages to block-oriented ones in general and from BPMN to BPEL, which helps to clarify significant existing achievements. An automated approach described in [49] for synthesizing a hierarchical SM [46] from the UML activity diagram that specifies a BP is referencing a stateful object to discover an object life-cycle.

Another approach [58] derives more precise process models by leveraging a process to the petri-net compiler, and producing most precise verifiable petri-net process model. A proposition of an algorithm extended petri-networks to the linear hybrid
automata is represented in [19].

The authors in [99] propose a model transformation mechanism from UML State Machines (UML-SM) to Discrete Event System Specification (DEVS) [131] in MDD context [29], through transformation rules implemented in Query/Views/Transformation (QVT) relations language in order to perform tools based simulations. An approach to the transformation of diagrams of UML-SM to a type of named Stochastic petri-nets was presented in [47].

Another work [38] where the authors made a transformation of UML-SM diagram to colored petri-nets by using Isabelle/HOL \(^8\). A plenty of work transform BPMN to YAWL in two different ways [55] [141] provided from BPMN to YAWL that mapped for each component from BPMN with an algorithm to transform them to YAWL processes. Another work [37] mentioned a transformation BPMN to YAWL by reusing its existing static analysis techniques to implement process-oriented applications.

An approach proposed to define a YAWL pattern for each BPEL activity in [23], that we can see that it is more efficient than BPEL. In [56], they made an analysis based on control-flow patterns of this transformation. The UML is also more referred to as the transformation field [60] because it captures various structural and computational aspects of behavior such as Activity Diagrams (AD) and SM.

A functional syntactic, vertical, and exogenous approach described to transform a BPEL program to abstract specifications is explained in [17]. Hypergraphs have been widely and deeply studied with different names and their instances, which can be found in [98] to solve problems that get emerged in areas such as propositional logic.

Next sections describe the several transformations examples done during our studies [89] [17].

\(^8\)Higher-Order Logic
3.6 Functional Approach To Transforming Abstract Specifications of FSM from BPEL Program

This work is one of our transformations approaches that we proposed after the comparison between models done, it revolves around the transformation rules which enable the construction of a moving structure between the program written in BPEL language to the descriptive elements of the PLC.

To achieve this objective, we have identified a set of situations (portions of BPEL code) that are recurring. These situations are modeled in the form of patterns. The proposed approach has been implemented in an environment that concertizes the implementation of patterns. The developed software tool also performs functions beneficial for protocol managers, such as BPEL code generators, verification, and visualization of some FSM properties.

In the context of services, the existing models represent different views (service protocols, composition, execution...) associated with the stages of the service life-cycle. It is clear that communities, designers, and users do not use the same models or languages; hence there is a need to find mechanisms to change from one model language to another. This can result in several of problems. Namely, it needs a new transformation method to abstract specification to facilitate interoperability and communication.

The approach proposed is very interesting in the field of Web services, and it is of paramount importance for the following reasons:

- Re-engineering of BP (re-building the model and BPR);
- Defining abstraction levels (model instead of language);
- Needs for evolution, development and maintenance;
- Existing of FSM verification software.

Whereas in different situations, the different situations, the management of BP for various reasons require transformations from and to models (re-engineering, maintenance, collaboration) and process languages trades. Several studies have largely dealt with the problem of model transformation as in table 3.3, but the exploration
of this literature shows that no study or approach has been proposed to allow the transformation from a BPEL code to FSM.

<table>
<thead>
<tr>
<th>Ref</th>
<th>Input</th>
<th>Output</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>[71]</td>
<td>BPMN</td>
<td>BPEL</td>
<td>Graphic aspects such as partitions and sub-partitions of BPMN models are lost.</td>
</tr>
<tr>
<td>[73]</td>
<td>UML</td>
<td>BPEL</td>
<td>Transformation does not generate BPEL artifacts: fault Handlers; event Handlers; Handler compensation; throw away; rethrow; compensate for; correlation set.</td>
</tr>
<tr>
<td>[9]</td>
<td>S-GNet$^9$</td>
<td>BPEL</td>
<td>Lacks some process element declaration details in the BPEL code obtained.</td>
</tr>
<tr>
<td>[4]</td>
<td>AUML$^{10}$</td>
<td>BPEL</td>
<td>Does not offer any formal structure to verify, validate and evaluate the system performance before the implementation phases.</td>
</tr>
<tr>
<td>[4]</td>
<td>BPEL</td>
<td>Pi-calcul</td>
<td>The pi-calculus of process algebra requires verification of protocol using logic and tools to correct and validate the model.</td>
</tr>
<tr>
<td>[71]</td>
<td>BPEL</td>
<td>TPn$^{11}$</td>
<td>The problem of blocking and liveness in petri-nets.</td>
</tr>
<tr>
<td>[48]</td>
<td>TPn</td>
<td>AHL$^{12}$</td>
<td>The problem of this work concerns performance measurements and verification of liveness properties as well as the analysis of reachability.</td>
</tr>
</tbody>
</table>

$^9$S-GNet: ScriptGen technology  
$^{10}$AUML: Agent Unified modelling Language  
$^{11}$TPn: Timed Petri-nets  
$^{12}$AHL: Algebraic High-Level
3.6.1 The Pattern-based Approach for transforming BPEL Program to FSM

The various characteristics of this approach of transformation from BPEL programs into abstract specifications of web services represented as below:

- The transition from source code to an abstract model;
- The transformation is exogenous;
- It is syntactic in the sense that it does not add semantics;
- It is vertical;
- It is based on patterns.

The transformation approach to be proposed will consist of a set of conversion rules, which allow to extract the target model’s structures.

First, we will define the general transformation rule, and we will give its formal specification. A generic transformation rule is a passage (or conversion) rule which makes it possible to take a portion of a source code identified by specific BPEL language primitives and to find the structure of the FSM corresponding to these primitives.

---

<table>
<thead>
<tr>
<th>Reference</th>
<th>Language</th>
<th>Model</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>[19]</td>
<td>UML</td>
<td>GSPN</td>
<td>The transformation tools used are not visual, and the resulting model after the transformation is not optimal and does not accept analysis and verification with tools analysis.</td>
</tr>
<tr>
<td>[67]</td>
<td>FSM</td>
<td>Hypergraph</td>
<td>Lack of FSM verification properties.</td>
</tr>
</tbody>
</table>

Table 3.3: Existing Transformations of FSM and BPEL.

---

13GSPN: Generalized Stochastic petri-nets
14JADE: Java Agent DEvelopment framework
CHAPTER 3. BUSINESS PROCESS MODELS

Second, we will identify a set of construction patterns where each will lead to a particular FSM target element. As we said this transformation approach uses the vertical transformation approach based on pattern matching. The list of patterns are:

\[ \text{Primitive} \rightarrow \text{PC} \rightarrow \text{Element} \] (3.1)

- **Primitive**: BPEL language primitives (sequence, invoke, receive, assign, switch, while, for, if–else, choice, loop).
- **Element**: Structure of FSM to be built (5-Uplet FSM \( = (M, Q, q_0, F, R) \)).
- **PC**: transformation rule applied if the conditions on the primitives are verified.

The semantics conveyed by this modelling is as follows: a structure named \textit{Element} is deduced from a BPEL program if the predicate: \( PC(\text{Primitive}, \text{Element}) \) evaluates to \textit{True}. This predicate is true if the constraints required by the element description are checked in the original part. In other words, the primitives of a BPEL program express the structure of the \textit{Element} to be built.

The identified patterns generate descriptive structures to specify the corresponding FSM. Seven patterns relating to control structures (loop, sequence, choice, etc.) have been identified and formalized for each. The logical expression allowing a proper evaluation of the predicate has been expressed.

3.6.2 Presentation of the Working Environment

While developing our application, achieving different functionalities needs different tools. We will present these tools as well as the reasons which motivated their choices.

- **Eclipse and Java**: As a programming environment we chose the Eclipse/Java couple. Eclipse is a Free Integrated Development Environment (IDE). It allows the development of programs in several languages: Java, PHP\(^\text{15}\). It uses the interesting principle of plugins, which allows adding functionalities according to needs. It is a contextual completion IDE\(^\text{16}\), it helps correct syntactic errors in code.

\(^\text{15}\)PHP: Hypertext Preprocessor, open-source server-side scripting language

\(^\text{16}\)IDE: Integrated Development Environment.
In our work, the external API\textsuperscript{17} used are:

- **JDOM\textsuperscript{18}**: is an API allowing the manipulation (\textit{creation, modification, deletion, ...}) of XML documents via a tree structure. It has some similarities to DOM. The reason that led us to use it is its simplicity and manipulation possibility with java.

- **Mysql-connector**: The API which allows access to databases under java is called JDBC\textsuperscript{19}. It is an executable Jar file (Java ARchive) that includes all the classes needed to access the database.

- **The XML Choice**: XML has become \textit{the lingua franca} in the IT world, a standard for representing structures of documents and data structures for information exchange. This XML success is largely due to its qualities, summarized in:
  
  - Strict separation between content and presentation;
  - Simplicity, universality, and scalability;
  - Text format with the management of special characters;
  - Strong structuring;
  - Document model;
  - Free format: ease of exporting an XML document in various other formats;
  - Existence of manipulation tools: \textit{JDOM, DOM}.

To verify the feasibility of the proposed approach, this section will present its implementation. The developed software prototype is briefly described, and some practical aspects are shown.

### 3.6.3 System Architecture

The \textit{BPEL-FSM} application (\textit{Transformation of BPEL programs into FSM 1.3}), which we implemented, contains seven modules that manipulate the information\textsuperscript{17}(API: Application Programming Interface, \textsuperscript{18}Java Document Object Model, \textsuperscript{19}JDBC: Java Database Connectivity.)
present in two databases. The following figure encapsulates a global view of the architecture of our system and the interaction between the various components.

The modules in figure 3.12 allow to perform different input/output, verification and obviously, the main function of transforming code into FSM.

1. The BPEL program generator module helps to create a complete program that can be modified, deleted using the BPEL program manager module.

2. External BPEL programs (imported) will be saved in the program database BPEL;

3. The generated BPEL programs will be saved in the BPEL programs database;

4. The old programs already existing in the BPEL program database is passed through the manager;

5. The syntax analyzer module uses the already existing BPEL programs to detect their anomalies;

6. The generated programs can go through the parser module;

7. The main module of this architecture which transforms the data recorded in the database BPEL programs;

8. BPEL programs transformed into protocols (FSM) will be recorded in the second database protocols;

9. The resulting protocols can be viewed;

10. Verification of FSM protocols is possible to detect errors if they exist;

11. If the FSM is not clear, the rectifier gives a hand to modify it manually, then saves it in the protocol database. In the rest of the section, we will explain the functionalities.

### 3.6.4 System Functionalities

The BPEL-FSM application 1.3 that we have implemented contains several functionalities allowing us to perform the following tasks:

1. **Generation of BPEL programs:** it is a graphical interface that allows you to
CHAPTER 3. BUSINESS PROCESS MODELS

Figure 3.12: System Architecture
create programs BPEL interactively.

2. **BPEL Program Manager:** The BPEL Program Manager is a necessary module in our application. It allows the manipulation of BPEL programs, and this manipulation is represented by this set of functions:

   - *Viewing BPEL programs:* is a function with the purpose displaying all the programs codes or a single program existing in the BPEL program database. Each display contains the names of the existing programs, the date of the last modification and the sizes.

   - *Updating programs:* a function ensuring the program’s internal modification. It can be: addition, deletion and renaming or the operation of a program.

3. **Verification:** this function allows to perform a syntactic analysis of BPEL programs and detect existing anomalies.

4. **The transformation into an FSM:** This module is the core of this work. It allows to produce a graphic PLC corresponding to the selected BPEL code (*full code*) as well as the corresponding PLC to a portion of code representing any activity (*flow, switch, sequence ...*). The following set of functions represents this transformation:

   - *Graphic display:* this function displays the PLC corresponding to a BPEL program selected.

   - *Graphic creation:* this is a function that allows you to create the business protocol.

5. **Protocol update:** this is a function for internal modification in the business protocol, which can be: add, rename and delete a state or an activity.

6. **Verification:** this is a function that makes it possible to perform a syntactic and formal analysis of the FSM (*a state not reachable, a state not reaching final states, absence of an initial or final state ...*).

   Another function is the transformation of the descriptive elements of the protocol (*states, transitions*) and to store them in the XML database. This protocol manager communicates with a protocol database for loading and saving.

   In our publication [17] a clear use case is represented with an example of transformation using the software implemented, which perfectly meets the objectives set and takes into account the conceptual aspects proposed in patterns section 3.1. The
purpose of the prototype produced is to extract the FSM corresponding to the BPEL program.

Several functions are also offered to the user. However, like any application in its first version, much remains to be done (automatic optimization of the display of the PLC, interface, control, . . . ). A functional approach based on a set of patterns that convert portions of BPEL programs into sub-structures of the target PLC to build was presented. The proposed method has been implemented in a software tool that produces the corresponding FSM PLC from an input BPEL program. The output result is presented in different forms (formal, XML, graphic), and verification actions are made to verify its correctness. Despite the limitation of the functionalities, the work plan is presented bellow:

• Improve the software tool and refine the graphic display of the deterministic FSM;
• Exploit a solid formalism which will take into account the problem of waiting;
• Study the possibility of considering not only the flow of activity but also the data conveyed by these activities during transformations. In this sense, given oriented process models (Date driven process) prove to be adequate to meet this challenge;
• Propose a model for handling exceptions and errors.

The next subsection will be about the remaining transformation or mapping between FSM and Hypergraph representation.

### 3.7 Hypergraph Representation Based on FSM

FSM and Hypergraph are completely different representations. In this context, our work [67] is a suggestion of an approach which aims to solve the problem of mapping between both models. We described a hypergraph to model all types of elements and patterns presented by FSM in that work.

As a start, we presented FSM elements with hypergraph elements. The Hypergraph is used to model diverse types of objects or features (states transitions). We suppose that every type of vertices corresponds to a state, and each kind of hyperedge
corresponds to the relation between them. Hyperedge can be labeled depending on transitions in FSM.

Secondly, the various FSM patterns represented by hyperedges can be divided into sub-sets of special patterns. Later, we will explain the different concepts in detail and describe theoretically how and what are the rules of mapping.

This suggested method takes advantages of the basic properties of generalized hypergraphs (Further explications will be presented in details in the context of next chapter).

Conclusion

In this chapter, we presented a detailed primary representation of existing BP models. We also conducted a comparison between these models to account out for positive and negative details within each model. In the end, we presented the transformation concept between different models and showed some examples of transformations done during our studies. The next chapter will investigate the Dynamic BP topic more in-depth.
Chapter 4

Dynamic Business Process Investigation

Introduction

The BP are related with both structure and functionality [119]; defined as a collection of activities or functions [67] put differently, BP are also defined as sequence of steps and activities taking one or more types of input and creates an output that is of value to the customer [89]. Both sides of BP; i.e. static and dynamic depend on the viewpoints and properties we are considering. The traditional approach used to implement a BP in today’s IS no longer covering the actual needs of the dynamically changing business [69][132].

4.1 Dynamic Business Processes

Processes have become an important asset for daily life in organizations because an adequate BPM (e.g. software development companies) helps to achieve organizational objectives. By nature, now BP are dynamic and influenced by a dynamically changing environment that obliged it to change under updating conditions at run-time. Over the years, approaches for supporting process change have received much attention from the research community because of the advanced development of hardware and software applied at different stages of the design process.
Several research efforts focused on problems of static BP but not on dynamic one. For that reason, we will be base our research this section on the concept of dynamic BP, recently crucial in changing effects. The use of BP can be considered diverse according to the view of the company or the structure of resources used. In this section, we will focus on the definition of dynamic BP.

From a general point of view, dynamic BP definitions refer to changes within the external and internal environment, and the consequences can be traced through adding, deleting, replacing components representing activities [104].

In this view, the business rules are formalized in order to condition the organization of the process and to guarantee that the result from its progress (at the end of the execution phase) is consistent with the business objective. In the table below 4.1 we collected all existing definitions and related concepts:

<table>
<thead>
<tr>
<th>References</th>
<th>Definitions</th>
</tr>
</thead>
<tbody>
<tr>
<td>[52]</td>
<td>Four main concepts are popular for defining the ability of a BP to adapt to the changing environment. These are: dynamicity, flexibility, agility and adaptability.</td>
</tr>
<tr>
<td>[95]</td>
<td>In order to create a dynamic model of a process, it is necessary to understand its functionality; the nature of dynamic modelling enables the representation of resources, time frames and the functionality of the individual activities contained within the process. Dynamic modelling enables a closer representation of the physical BP environment including people and equipment, resources and their movements.</td>
</tr>
<tr>
<td>Reference</td>
<td>Description</td>
</tr>
<tr>
<td>-----------</td>
<td>-------------</td>
</tr>
<tr>
<td>[30]</td>
<td>Not all authors define the concept of dynamic process clearly, Ad-hoc processes are always used together with flexible and dynamic processes and mostly used with adaptive processes, but never with variable processes. Variable processes are always used together with flexible and dynamic process and mostly with adaptive process. Adaptive processes are always used together with dynamic process and mostly with flexible processes. Flexible process is always used together with dynamic processes.</td>
</tr>
<tr>
<td>[97]</td>
<td>Dynamic BP is a variant of agile process and enables a business user to make changes in the process at run-time.</td>
</tr>
<tr>
<td>[64]</td>
<td>Dynamic BP is not defined strictly at the beginning of its execution and it changes under new conditions at run-time.</td>
</tr>
<tr>
<td>[124]</td>
<td>Dynamic BP can implement a BP model whose components (a set of conditions, a set of activities, a content of activity, a set of activity sequences, a set of decision nodes, the participants) may vary and, if necessary, change with low latency at run-time due to changes of the context).</td>
</tr>
<tr>
<td>[133]</td>
<td>The dynamics of BP is characterized by a variety of terms: ad-hoc (there is no predefined definition), adaptive (the ability of workflow to react to exceptional circumstances), dynamic (must have flexible and adaptive execution), flexible (ability of a process to adapt to the changes in the environment or to its changing requirement) and variable process (adapted as a response to evolution in internal and external environment).</td>
</tr>
</tbody>
</table>
CHAPTER 4. DYNAMIC BUSINESS PROCESS INVESTIGATION

Dynamic BP where content and the sequence of activities depend on the context of the environment and can be changed at run-time. In contrast with dynamic BP, static BP has a strict specification; i.e. the content and sequence of activities are defined before BP instance execution and cannot be changed at run-time.

The concept of dynamic BP can be perceived as a group of states. A collection of BP activities might change at a certain point in time due to the changes occurring in the BP context, therefore, a sequence of activity execution cannot be predefined in advance.

Table 4.1: Definitions related with dynamic BP.

| 4.1. Definition. Dynamic BP is the process that can change some BP activities during run-time (at certain points) under various conditions that are not predefined in the beginning, it can react and adapt from internal or external environment changes, it is flexible and agile in its environment to express the concept of BP dynamicity. All properties of dynamic BP concept (adaptive, variable, flexible, Ad-hoc and dynamic) are related and used together with various relation degrees in order to respond to different scenarios. |
| 4.2 Dynamic Business Process Management |

Traditional process management does not encompass the majority of processes in modern organizations, Dynamic BPM is an extension of traditional BPM. Unfortunately, due to the competitive environment and the massive technological changes the aforementioned threats negate at an increasing pace the benefits which stem from implementing traditional process management.

There are numerous definitions related to BP dynamics. Previously, we postulated some of these definitions 4.1 to finally come up with an acceptable and a reliable definition. The concept dynamic BP definitions might be postulated as follow:

4.2 Dynamic Business Process Management
At a time when the competition is merely “a click away,” organizations are challenged by the necessity of adapting their fundamental BP to the requirements of individual clients. And because the expectations, habits, and capabilities of the clients are diverse, or even conflicting, the key to success no longer lies in the optimal, “averaged” BP, but in the most skillful dynamic structuring of BP.

Due to the necessity of maintaining processes that satisfies client needs, as well as adapting process performance to ongoing changes and specific contexts of performance, it becomes impossible for the process owner or manager to introduce changes on an ongoing basis. In various enterprises, it is crucial to empower direct process performers with the capability to dynamically modify process performance. According to [121] Dynamic BPM defined as below:

4.2. Definition. Dynamic BPM is understood as management enabling organizations to react to fundamental conditions of operation (both internal and external) and cater to the client individual needs in a timely fashion (and in the case of critical factors—practically instantaneously) on the basis of process adaptations entered in real time in the course of performance itself by their direct performers.

In accordance with the correspondence principle, the concept of dynamic BPM does not replace traditional (static) BPM as such, as much as it is its expansion and extension, that is, allowing us to understanding and the description (accurately anticipate or manage) a broader part of reality than traditional management.

4.3 Factors Impacting Dynamic BP

The Management of the BP evolution [2] require the good comprehension of the different changes which are caused directly from the application of various factors (one or more) on different process elements [69]). These changes could be caused during addition or the deletion of some elements or activities.

There are several reasons that require process change, including requirements derived from new business strategies, new policies, new laws and new applications [129] [136] [94] [3], problems with the original process, BP re-engineering and process enhancement needs [127].
Enterprises either commercial, businesses or government organizations are recently encounter a range of challenges. These challenges and different factors are impacting their components during BP execution (run-time).

In fact, the success of enterprises highly depends on their ability to react to changing conditions of operation in their environment in a quick and flexible manner [1] which leads to the internal and external impact of the environment and its nature on BP during execution time.

Changes of the environment, like new government regulations, stock price change, or internal changes like business volatility, desire to remain competitive, motivate companies to change BP components quickly.

In [142] the authors in this context examine the impact of internal changes on BP and the capability of processes to adapt themselves to the changing environment; the externally initiated alterations are not considered. Dynamics [75] is dependent on internal factors (Business strategy, organizations infrastructure, IT) and external (market, regulating acts by government, changes in technologies drivers).

Resources as well are able to impact changes on the workflow. Limited resources generate problems during workflow execution which can block the working of the process or impact the capability to change few BP characteristic. Resources that appear on workflow structures are: sequential, parallel, choice and loop patterns. In [75] only sequential and parallel patterns. The processes should be capable for dynamic adaptation [130] to different scenarios, although the method for adaptation cannot be exhibited in detail.

In reference [111] we found one offline stage which determines the required resources to execute a workflow while minimizing the cost and meeting the user specified deadline. A second online stage reacts to structural changes taking place at run-time like adding tasks during execution.

Next section we will search and present the components impacted during the execution time.
CHAPTER 4. DYNAMIC BUSINESS PROCESS INVESTIGATION

4.4 Process Components that Change at Run-time

In this section we will investigate the process components that changing under the impact of different factors, after that we will collect some important components at the execution time (i.e. during, before, or after run-time).

*The Process Activity:* as we know activities are the main components of BP and they have the responsibility to handle different events of BP that may be running in cases, namely defined or non-defined *(formed at run-time).* These activities are those ones affected during changes; an activity can be modified, a new activity can be added or existing one can be removed.

Some previous works implement dynamic BP tools *(DYPROTO)* [139] inserting new activities, deleting old ones or dynamically implementing activity *loop*, another tool [145] which is flexible adapting to business environment changes and determines the sequence of activities at BP run-time. But these solutions don’t meet the dynamic BP requirements.

When changes alter the workflow; i.e. BP during their run-time which is seen as the composition of a set of basic operations *(general flow structures including sequential, parallel, choice, and loop patterns)* that must be performed on a given input data set to produce the expected result. So, these modifications can provide data and make some changes on it. This problem is solved by could computing emerging technology which offers a new model of service provisioning for workflows applications. It provides, on demand network access to a shared pool of computing resources that can be elastically provisioned and released. Then, the elastic nature of cloud environment enables such dynamic workflow to be enacted more efficiently since it facilitates the changing of resource quantities at run-time.

In our first paper [89], we propose an approach where the data and business structure *(operational level and design level)* is expanded with an analysis phase *(organizational and strategic planning level)* to observe and to detect business events that enforce changes in process flow, documents and document flow. The impact of business events influences both structure of the workflow and the structure of document types.

The several challenges impact the architecture of enterprises, and environment as...
well such as changes in economy, society, physical environment, economics, culture and politics. For these reason, enterprises should be able to focus their attention on all those impacts to react flexibly toward the stimuli appearing the form of business events. The proposal method is an extension of the BP modelling approaches with an organization and planning level according to Zachman Framework [144] thereby an “analysis season” is created beside the “design season” and “operational season” [10].

The proposed framework is exhibited in figure 4.1 [89]. The analysis level provides the chances to observe and to detect business events that enforce changes in process flow, documents and document flow. The impact of business events influence both structure of the workflow and the structure of document types.

![Figure 4.1: The Architectural Building Blocks for Unified Modelling Framework.](image)

The representation of BP either in petri-net or BPMN can be perceived as a document in XML format [14]. The documents that are data collections, and the process models can be represented using the concept of document types. The document types can be described meticulously in a hypergraph. The hyperedges can depict both internal structure of document and BP models. Predicates of description logic and rules can be attached to the intentional documents and to the relationships.
between BP model, document types and documents. The hypergraph and related
mathematical tools provide the chance to keep and to enforce the consistency, integrity
and compliance of models.

Next, the process context where the set of external and internal factors and
conditions changes is related to the development of a new promising paradigm for
BP modelling to support explicit definition of the "context related knowledge" that
should be identified. For example, in health process activities are selected according
to the user’s context, which describes user’s health state. When a user’s context
changes, like blood pressure or temperature, necessary activities,

It is difficult or impossible to predict all possible changes of a context and
predefined possible sequences of activities. Contrary to dynamic BP, static processes
have predefined sequence of activities and could be adopted only in predefined places
of a process instance, like variation points.

Researchers argue that any information reflecting changing in circumstances
during the execution of a BP can be considered as contextual information. The
context is thus defined as “the collection of implicit assumptions that is required
to activate accurate assignments in the BP model based on the process instance
level”. The role-driven [108] BP modelling approach (RBPMo) presented in [107] was
extended to support context awareness.

4.5 Dynamic BP Requirements

Most recently, the rapidly changing business environment and thereby originated
adaptive requirement against business IS led to the agile business approach as a
management science philosophy. In this section we will have a look and represent
dynamic BP requirements. Dynamic BP should satisfy as below:

- **Dynamic BP have no predefined sequences of activities**: Activities are
  not predefined but formed at run-time: so every subsequent should be selected
  according to the external and internal context and predefined rules. Therefore,
  every subsequent instance may differ from the other instance of the same BP, if
  there is no activity for further execution at a dynamic BP run-time, we should
  terminate the execution of a dynamic BP instance or define a new activity and
concerning rules for a BP instance execution.

- **Dynamic Context-based**: BP must support changes due to any process context [53] which is defined with a set of external (variables and rules) and internal (state of system and resources users) factors and dynamic BP should react with that context changes.

- **Rule-based Dynamicity**: it must define new business rules, change/delete existing ones [64] at BP run-time. Dynamic BP should be reacting to these changes formulated at run-time of the business rules at process instance. Every next activity in the sequence within dynamic BP should be selected according to the predefined rules.

- **The Low Latency**: Dynamic BP execution should support dynamic BP instance change, according to role of the process, at any time, with very low latency compared with duration of the process.

- **Analyzing of Historical Data of Instance**: Before selecting the next activity, the historical data (stored in a log file) of instances execution (can be a good practice which is suitable instance from the historical data for repeated execution, or as bad practice should not be executed) of the same dynamic BP should be analyzed and the selected next activity should not cause execution of an unacceptable sequence of activities. In addition time, cost, and values should be calculated and stored for each executed dynamic BP instance. This is a requirement before describes that before executing any dynamic BP instance.

- **Supporting of changes**: BP should support changes to any process component (a set of conditions, activity, activity sequences, decision nodes and participants).

The table below 4.2 summarizes requirements and their role:

<table>
<thead>
<tr>
<th>Requirements</th>
<th>The Role</th>
</tr>
</thead>
<tbody>
<tr>
<td>No predefined sequences of activities</td>
<td>The advantage of implementing this requirement is that we increase dynamicity of a process.</td>
</tr>
</tbody>
</table>
### Chapter 4. Dynamic Business Process Investigation

**Rule-based dynamicity**

The main opportunities of using business rules to ensure dynamicity of BP are: each activity in a process is selected according to the defined conditions (*expressing constraints of application domain*) at BP runtime, and the content of an activity is chosen based on the changing internal and/or external context, and it should be implementing those changes at process instance runtime.

**Low latency**

The advantage of implementing this requirement is that the change of a context or of business rules influences selection of the next activity immediately after the changes comes in force at process instance changes and allows rising process dynamically.

**Analyzing of previous activity (historical data)**

The analysis of historical data allows determining the so called "good" (*requires minimum resources and process is reached*) and "bad" (*use too many resources*). During historical data analysis, we can define what conditions i.e. state of current context and propose suitable alternative sequence of activities for execution.

**Supporting of changes to any process component**

Components and their relationships are organized in such a way as to support rule and context-based dynamic BP activity in modelling and simulation. They are also included in architecture adding intelligent functionality.

<table>
<thead>
<tr>
<th>Requirement</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Rule-based dynamicity</td>
<td>The main opportunities of using business rules to ensure dynamicity of BP are: each activity in a process is selected according to the defined conditions (<em>expressing constraints of application domain</em>) at BP runtime, and the content of an activity is chosen based on the changing internal and/or external context, and it should be implementing those changes at process instance runtime.</td>
</tr>
<tr>
<td>Low latency</td>
<td>The advantage of implementing this requirement is that the change of a context or of business rules influences selection of the next activity immediately after the changes comes in force at process instance changes and allows rising process dynamically.</td>
</tr>
<tr>
<td>Analyzing of previous activity (historical data)</td>
<td>The analysis of historical data allows determining the so called &quot;good&quot; (<em>requires minimum resources and process is reached</em>) and &quot;bad&quot; (<em>use too many resources</em>). During historical data analysis, we can define what conditions i.e. state of current context and propose suitable alternative sequence of activities for execution.</td>
</tr>
<tr>
<td>Supporting of changes to any process component</td>
<td>Components and their relationships are organized in such a way as to support rule and context-based dynamic BP activity in modelling and simulation. They are also included in architecture adding intelligent functionality.</td>
</tr>
</tbody>
</table>

*Table 4.2: Dynamic BP Requirements and their Roles.*
CHAPTER 4. DYNAMIC BUSINESS PROCESS INVESTIGATION

The representation at business analysis and process level should consider the demands for changes of the data perspective. The requirement for changes at data perspective can be perceived as modification in the structure of documents, collections of data and database schema. The intimate interrelationship between documents and BP can assist to deduce the requirements for changes dynamically.

The traditional linear model that involves design, builds independent test, implements and supports design, builds and tests occurs iteratively, and theoretical requirements are not system requirements, but BP requirements. A more traditional private sector organizational model should be used; i.e., one that explicitly addresses the importance of BP requirements in defining an Enterprise Resource Planning (ERP) implementation. The role of BP requirements as mentioned in [16] is to support of the life-cycle implementation process, support BP functionality described in the models, and to facilitate analysis and BP configuration management.

As conclusion for this section we can say that: each dynamic BP requirement has an important role when we implement. Therefore, we should observe these requirements when we want to propose an approach or model for dynamic environment and these requirements can guide the selection of activities.

4.6 Limitations Exist in Current Research

This domain research has many limitations listed as follow:

- There is no research leads to possible dynamic measures.
- There is no research on concurrently running dynamic BP, when they are using different type of resources in concurrent manner.
- There is no full-scale research on impact of internal and external contexts on running dynamic BP.

4.7 Measures Related to Dynamic BP Quantification

The fact that Performance Measurement Systems (PMS) [63] need to achieve alignment with strategic priorities is well established within the performance measurement
literature [26][39] [90]. However, also commonly recognized that the external and internal environment of an organization is not static but is constantly changing. The Integrated PMS audits identified that PMS needs to be dynamic by:

- Being sensitive to changes in the external and internal environment of an organization;
- Reviewing and reprioritizing internal objectives when the changes in the external and internal environment are significant enough;
- Deploying the changes to internal objectives and priorities to critical parts of the organization, thus ensuring alignment at all times;
- Ensuring that gains achieved through improvement programs are maintained.

Therefore, a dynamic PMS should have:

- An external monitoring system, which continuously monitors developments and changes in the external environment;
- An internal monitoring system, which continuously monitors developments and changes in the internal environment and raises warning and action signals when certain performance limits and thresholds are reached;
- A review system, which uses the information provided by the internal and external monitors and the objectives and priorities set by higher level systems, to decide internal objectives and priorities;
- An internal deployment system to deploy the revised objectives and priorities to critical parts of the system.

However, the reality is more complex. In practice, there may be a seldom event, which may cause the whole organization to review its corporate-level objectives and priorities, which results in the need for restructuring the whole PMS. It is more likely that changes within the immediate environment of a business unit or a BP may affect the way that business unit or process could best contribute to the organization’s overall objectives. That is, the need for change is not always driven from the very top of the organization but more frequently it is initiated as a result of an external or internal change within the immediate environment of a business unit [15] or BP.
CHAPTER 4. DYNAMIC BUSINESS PROCESS INVESTIGATION

4.8 Facilities for Quantification and Modelling Dynamic BP?

For Static modelling

As previously explained static modelling simplifies a representation of BP at a point in time. As such, they ignore dynamic behavior. In fact, static models are very useful in understanding and representing the structural features of BP and can be valuable means of communication. However, BP often display complex interactions [14] that can only be understood by unfolding behavior through time.

Numerous methods exist for the static modelling of BP including Role Activity Diagrams, CSP, petri-nets, flowcharts to name but a few. However, deployment of these formal techniques is very limited, a more easily understandable method, similar to that of flowchart, is being favored. It is possible to claim that this is indeed a static model of the existing Service Ordering Process. Using this reveals the possibility to determine the basic functionality of the processes but apart from that it is difficult to derive anything further. However, this does not mean that it is impossible to design a changed, improved, version of the process.

It is possible, by simply glancing at the changed process, to notice that the extant process has been simplified quite considerably. On the surface, this appears to be a feasible alternative. However, the impact of the process cannot be determined without implementing it within the physical environment.

Furthermore, it is difficult to establish the resources allocated to this process, whether any of the activities work in parallel, the length of time required to complete each activity and time-frame for the process from starting till finishing.

For Dynamic Modelling

The dynamic BP behavior may change over time as a result of resource competition, interactions, or other sources of uncertainty. Dynamic modelling is about enabling the outcome of a change process to be evaluated prior to its implementation into the physical environment.

Modern enterprises and organizations operate in a dynamic environment [21] that
CHAPTER 4. DYNAMIC BUSINESS PROCESS INVESTIGATION

is constantly evolving, so it must adapt its processes and establish a framework for modelling and analyzing workflow processes. From the table shown before 4.1 we can conclude a simple Dynamic BP modelling definition:

4.3. Definition. Dynamic BP modelling is the representation of a BP linked with time, the model includes activities, components, functionalities, resources, individuals (participant) that need to be understood before modelling, so the result of the modelling exercise could be implemented and simulated.

The main opportunities for using business rules to ensure dynamic aspect of BP (answer of the research question 4) lie in the following:

Each activity in a process is selected according to the defined conditions at BP run-time; choice of activity content; and representation of changing dynamic BP context. However, and as presented there is no complete approach or tool for rule and context-based dynamic BP modelling and simulation i.e. none of the analyzed tools which are widely used for BP modelling and simulation, support changing business rules during the simulation of dynamic BP. Some approaches, like [59], describe BP dynamicity using BP point-cuts, where adaptations can be made, or changes of BP are available in new instances of the BP, but not at the same instance, like in [140].

Process adaptation is a topic that raises interest in the research community, however, there is still no integration. In [59] the authors presented Complex Event Processing (CEP) for Context-adaptive processes in pervasive and Heterogeneous Environments, a framework that intends to facilitate the integration of CEP into existing BP and to allow these processes to be dynamically adapted to different circumstances and to address four issues: adaptation, dynamicity, integration to BP, and non-dependency to a specific CEP engine. As part of the framework proposed the Standard BP Language (SBPL), an extension of BPEL allows the user to include the adaptation points and conditions in order to create dynamically adaptable BP.

It is necessary to understand process functionality; the nature of dynamic modelling enables the representation of resources, time-frames and the functionality of the individual activities contained within the process. Dynamic modelling enables a closer representation of the physical BP environment including people and equipment; resources [95] and their movements.
Conclusion

In this chapter, we presented and investigated deeply our thesis topic, dynamic BP definitions were presented from several and famous references in the domain, our own definition was defined as well 1.3. Different factors and their impacts were explained and investigated during our studies.

The next chapter will be about the concept of the hypergraph and its representation and the hypergraph based on the FSM approach. A formal verification will be applied to verify several properties.
Chapter 5

Hypergraphs and Formal Verification

Introduction

Hypergraphs as representational structures are apt to system modelling, analysis and specification, exclusive to information and software systems. Formal tools consistency checking are available by test algorithms.

This chapter provides an overview on directed graphs and hypergraph. We will present some essential definitions and basic concepts of hypergraph and application domains, followed by an implementation made during our investigation for the hypergraph using Python. Formal Methods and formal verification objectives will be defined with particular attention to model checking. In the next section, the hypergraph based on FSM approach will be presented and several properties will be verified at the end of this chapter.

5.1 Hypergraph Theory and Concepts

Unlike other mathematical theories of the 20th century, hypergraph theory is one of the fresh independent theories proposed to generalize the theory of graph. The 1960s witnessed the birth of this hypergraph theory in France (France school) by C. Berge, and the leaders of mathematics from Hungary: Paul Erdős, László Lovász, Paul Turán. . . . It was the first generalization of hypergraph concepts, it was a generalization of concept of graphs into hypergraph they extended, as well, the concept of edge into
CHAPTER 5. HYPERGRAPHS AND FORMAL VERIFICATION

(hyperedge).

A hypergraph is a structure that describes complex relationships that can be explored among models during analysis and design of IS; it is a generalized graph theory that plays a vital role in discrete mathematics [27]. We start with its basic definitions:

Hypergraphs generalize the notion of a graph by defining hyperedges which contain families of vertices, unlike conventional edges which join only two vertices. From a theoretical point of view, hypergraphs make it possible to generalize certain theorems of graphs, or even to factor several of them into one. From a practical point of view, they are sometimes referred to as graphs since they better model certain types of constraints. In this section, we present some essential terms of hypergraphs.

5.1. Definition. Hypergraph $H$ is pair $(V, E)$ of a finite set of $V = v_1, v_2, \ldots, v_n$ and a set $E$. The elements of $V$ are called vertices (nodes) without repetition and the elements of $E$ are called edges which are a subset of $V$ [22] where:

1. $E_i \neq \emptyset$; $(i = 1, 2, 3, \ldots, m)$;
2. $\bigcup_i^m E_i = V$.

Example: The example below 5.1 shows the structure of hypergraph were $E = \{e_1; e_2; e_3\}$ are edges and $V = \{v_1; v_2; v_3; v_4; v_5\}$ are vertices (nodes):

![Figure 5.1: Hypergraph Structure.](image)

Formally, let $G(V, E, w)$ denotes a hypergraph [78], where: $V$ denotes a finite set of nodes $V$, $E$ denotes the set of hyperedges $e$, $w$ is a weight function defined as $w : E \to R$. Each hyperedge $e \in E$ is a subset of $V$ and is assigned a positive weight $w(e)$.
5.2. Definition. We say hypergraph of order \( n \) if \( |V| = n \) and the size of a hypergraph is equal to the number of occurrences of the vertices in its hyperedges.

Example: Figure 5.2 illustrates a hypergraph \( H = (V,E) \) of order 8 and size 15 such as:
\[
V = \{1, 2, 3, 4, 5, 6, 7, 8\} \quad \text{and} \quad E = \{\{1, 2\}, \{2, 3, 7\}, \{3, 4, 5\}, \{4, 6\}, \{6, 7, 8\}, \{7\}\}.
\]

![Hypergraph Example](image)

Figure 5.2: Hypergraph Example.

A generalized hypergraph could be defined as:

5.3. Definition. A generalized or extended hypergraph \([12]\) is that some of the hyperedges are denoted in certain cases– as vertices, thereby a generalized hyperedge \( e \) may consist of both vertices and hyperedges as well. The hyperedges that are contained within the hyperedge \( e \) should be different from hyperedges \( e \).

In defining the concept of directed hyperedge or hyperarcs \([88]\):

5.4. Definition. A hyperarc is an ordered pair \((\text{From, To})\) \( E = (X,Y) \) of (possibly empty) disjoint subsets of vertices; \( X \) is the tail \( T(E) \) of \( E \) while \( Y \) is the head \( H(E) \). The concept of the directed hypergraphs is an ordered pair of vertices and hyperarcs that are directed hyperedges, i.e. each hyperarc is an ordered pair that contains a tail and a head.

The domain determines the term Forward-b (BF-hypergraph) \([81]\) as:

5.5. Definition. A forward hyperarc (F-arc) is a hyperarc \( E = (T(E), H(E)) \), with \( |T(E)| = 1 \). A backward hyperarc (B-arc), is a hyperarc \( E = (T(E), H(E)) \) with \( |H(E)| = 1 \). A BF-hypergraph is a hypergraph whose hyperarcs are either B-arcs or F-arcs.
CHAPTER 5. HYPERGRAPHS AND FORMAL VERIFICATION

We can define the concept of directed path [5] or hyperpath as follow:

**5.6. Definition.** A directed path \( P_{sd} \) from a source \( s \) to a destination \( d \) in \( H = (V, E) \) is a sequence of nodes and hyperarc \( P_{sd} = v_1 = s, E_{i_1}, v_1, E_{i_2}, \ldots, E_{i_q}, E_{q+1} = d \) where: \( s \in T(E_{i_1}), t \in H(E_{i_q}). P_{sd} \) is said to be a cycle, when \( t = s \).

All the previous concepts defined are used and applied in different domains and various studies. Many concepts in the theory of graphs, may be applied to hypergraphs, as many theorems that are applicable to graphs.

There is a countless number of concepts and terms related to hypergraph like directed colored multigraph where each edge is colored by one of the \( k \) different colors, minimum st-hypercut: an st-cut that uses the minimum number of sequence hyperedges, and the maximum st-hyperflow which represents maximum number of hyperedge-disjoint st-hyperpaths. All those concepts are not used in our research but we just mention them for the task of deep research in the domain. The next section presents several uses and applications of our above tackled concept.

### 5.2 Hypergraph Application Domains

In the last years, hypergraph is largely applied in various studies and in different domains. In this section, an introduction of these several domains is provided to explain more the importance of the hypergraph.

Hypergraph learning has achieved a consideration performance in many implementations and areas. Several authors [5] expressed hypergraph for clustering by using a clique average to transform it into a simple graph.

In addition, hypergraph as well used in the domain of data structure [109], multiple learning [120] and video segmentation [41], they are used to solve many difficulties in the field of images processing [101], classification and to connect the high order of relationships and samples, where samples are vertices and edges are similarity between two ones.

Another researcher [123], suggested a semi-supervised learning method called Hyper-Prior for group factor expression data by using biological knowledge as a constraint.

89
In [41], the researcher involves the task of image clustering as an issue of hypergraph partition. Each image and its nearest neighbors form two kinds of hyperedges based on the descriptors of shape or appearance. In another work [86], a hypergraph-based image retrieval approach is proposed, where hypergraph-based 3-D object recognition is proposed.

The big importance and the huge application areas of hypergraph representation solve different issues, thus its importance:

- Hypergraph model has a strong theory for solving real world problems;
- Hypergraph could be networks model because of its mathematical tools;
- Hypergraph generalizes theorems on the graph.
- Hypergraphs are an appropriate structure for modelling, examining and specifying systems;
- Its formal tools consistency checking is available by test algorithms;
- It is able to describe complex relationships during the design and analysis phase of the IS;
- Hypergraphs can be grasped in their capabilities to describe heterogeneous finite structures.

5.3 Hypergraph Representation

BP modelling is mainly used to map a workflow so we can understand, analyze, and make positive changes to that workflow or process. The usage of diagram visualizations helps us to understand this process and make better decisions.

Our experiments [82] can be represented as three dedicated and independent approaches. First, two approaches came up, Amazon Web Services (AWS) services and wrapped up with Python programming language. The last was assumed as pure python implementation with HyperNetX \(^1\), NetworkX \(^2\) and Pandas\(^3\), external libraries.

\(^1\)HyperNetX library: https://github.com/pnnl/HyperNetX
\(^2\)NetworkX library official documentation: https://networkx.github.io/
\(^3\)Pandas library official documentation: https://pandas.pydata.org/
CHAPTER 5. HYPERGRAPHS AND FORMAL VERIFICATION

Approaching AWS services we realized it was a too costly decision, which invoked the reason to swap to the last approach with manual generating data and the rest activities. Beyond, it was conceived AWS allows us to handle a very wide range of tasks, but we needed to focus on more tidy problem definition. We concluded most of the known models are outdated and are not as flexible as we needed. However, it must be highlighted that some of them are used far more effectively in other industries and for different purposes. Because of that, we considered focusing on simulating within the bipartite graph and hypergraph representations.

5.3.1 AWS Services

We decided to use multiple AWS services as the main components because they allow us to increase the time-box for achieving results. AWS is a cloud computing service and the world’s most comprehensive and broadly adopted cloud platform, offering over 175 fully-featured services from data centers globally. While experimenting, we used the following services:

- S3: simple storage service that offers industry-leading scalability, data availability, security, and performance.
- Lambda-service: which lets us run code without provisioning or managing servers.
- DynamoDB-key-value and document database: delivers single-digit millisecond performance at any scale.
- Neptune-fast and reliable, fully-managed graph database service: makes it easy to build and run applications that work with highly connected datasets.
- Redshift-fully managed petabyte-scale data warehouse service in the cloud.

AWS S3: official Amazon Cloud Services documentation: https://aws.amazon.com/s3/
AWS Lambda, official Amazon Cloud Services documentation: https://aws.amazon.com/lambda/
AWS DynamoDB, official Amazon Cloud Services documentation: https://aws.amazon.com/dynamodb/
AWS Neptune, official Amazon Cloud Services documentation: https://aws.amazon.com/neptune/
AWS Redshift, official Amazon Cloud Services documentation: https://aws.amazon.com/redshift
• **DynamoDB + Cache + Titan** - Amazon DynamoDB Storage Backend for Titan, enabling to store Titan graphs of any size in fully-managed DynamoDB tables. Graph databases are optimized for fast traversal of complex relationships required for social networks, recommendation engines, fraud detection, inventory management, and more. Titan is a popular graph database designed to efficiently store and traverse both small and large graphs up to hundreds of billions of vertices and edges.

• **CloudWatch** - monitoring and observability service built for DevOps engineers.  

![Figure 5.3: AWS Services.](image)

5.3.2 **AWS CloudWatch**

CloudWatch stood as one of the main components within our experiments, because it collects monitoring and operational data in the form of logs, metrics, and events, and visualizes it using automated dashboards so we can get a unified view of the AWS resources, applications, and services that run in AWS and on-premises. It can correlate the metrics and logs to better understand the health and performance of the resources.

We can also create alarms based on metric value thresholds to specify, or to watch for anomalous metric behavior based on machine learning algorithms. To take action quickly, we can set up automated actions to notify us if an alarm is triggered and automatically start auto scaling, for example, to help reduce mean-time-to-resolution.

---

9AWS CloudWatch, official Amazon Cloud Services documentation: [https://aws.amazon.com/cloudwatch/](https://aws.amazon.com/cloudwatch/)
It can also dive deep and analyze our metrics, logs, and traces, to better understand how to improve application performance.

Meanwhile, our main task was to align with the bipartite graph and hypergraph, that is why we needed to move further within additional tools. So, let us consider the three main approaches we proceeded with.

5.3.3 The First Approach

The first and second approaches schema are presented in the diagram below.

First of all, we partially set up these tools as a pipeline. The key was to obtain first the "environment" and observations immediately. Therefore, we decided to focus not on creating a complicated chain of tools, but we tried to obtain that setup as soon as possible. We have created a script that will parse data from 3rd party API (as a source of data we decided to take the London transportation API\textsuperscript{10} as we mentioned before). We obtained data about car accidents for different years. Here we solely focused on the process, not on data itself.

After passing the data to S3 under a dedicated bucket and with a specified key, AWS Lambdas were automatically triggered to process and forward data to DynamoDB and Redshift in parallel. We decided to use two different databases because, firstly, we considered storing processed data and logs separately, and we wanted to create processes similar to the real work pipelines.

We aligned the processing steps because the format of input data from TFL

\textsuperscript{10}Transport for London API official documentation: \url{https://tfl.gov.uk/}
was in JSON format. We converted it into "csv"\footnote{CSV: Comma Separated Value} for further convenience to interact with DBs. In the meanwhile, all the used AWS components were tracked by CloudWatch. We monitored it also. Once the data was passed to DynamoDB we prepared pure Python functionality to grep data from CloudWatch to analyze the features of processes as well as to explore and define CloudWatch Custom Metrics.

The main complexity within DynamoDB is found in Titan App and Elastic Cache. Titan and the others are not favored, and they are not maintained properly, moreover, it lacks documentation. Based on that, we decided to try, on one hand, to simplify our chain of tools but on another to keep the complexity of processes. The second approach came up here.

5.3.4 The Second Approach

The first few steps remained unchanged. Needless to say we found that few other AWS services fit our purpose best by the provided documentation. Here, the main difference is in AWS Neptune and Gremlin that allows obtaining from out of the box, graph-based features, and language (Gremlin) to query that data from Neptun.

We established a specified scheme in the documentation, but once launched that chain of tools, we received an alarm immediately (which we have set up beforehand for such kind of expected cases). We would like to point out that AWS Neptun is the...
CHAPTER 5. HYPERGRAPHS AND FORMAL VERIFICATION

best service out of the box to store and manage graph-based data.

After all, we were looking at the most time-efficient methods of solving our task. Based on that, we manually generating data, then we applied open-source tools to analyze this data and pass into a bipartite graph and further into a hypergraph, then we tried to compute the performance of the finally collected observations.

5.3.5 The Third Approach

The third approach schema is presented in the following diagram:

![Diagram](Figure 5.6: Third Approach Schema.)

After exploring we assumed following types for process type data: bulk; load; download; remove; create. For activities types we considered: multiple merge; multiple choice; parallel; exclusive choice; sequence; cycle. Finally we defined the next characteristics for process features: process id; process type; process cost; execution time; user role. These are the steps followed:

- Generate simulated BP data;
- Pass generated data into a bipartite graph;
- Pass BP data into hypergraph;
- Apply Smith Normal Form.

**Generate simulated BP data**

We defined process types and activity types that stand for interactions inside of process types and process feature additionally. The figure 5.7 shows more details as follows:
CHAPTER 5. HYPERGRAPHS AND FORMAL VERIFICATION

Figure 5.7: Generate Simulated BP Data.

From Bipartite graph to Hypergraph

After we generated data, we forwarded it into a bipartite graph. as in figure 5.8:

The reason for that is to have more convenient basis to form hypergraph itself from obtained bipartite graph which will be more convenient to form a hypergraph itself from the obtained bipartite graph. In this context, we relied on two two dedicated functions. We first handled input (previously generated data) and created a bipartite graph, and then we cast a bipartite graph into a hypergraph.
Afterward, we applied the Smith Normal Form to find similar processes groups and make further computations.

Figure 5.9: Smith Normal Form.

In mathematics, the Smith normal form \(^{12}\) is a normal form that defined for any matrix with entries in a principal ideal domain (PID), in a simplistic way integers.

An incidence matrix is a matrix that shows the relationship between two classes of objects. If the first class is \(X\) and the second is \(Y\), the matrix has one row for each element of \(X\) and one column for each element of \(Y\). The purpose of computing the homology [40] groups for data generated hypergraphs is to identify data sources that correspond to interesting features in the topology of the hypergraph.

The elements of one of these homology groups are generated by \(k\) dimensional cycles of relationships in the original data that are not bound together by higher-order relationships. Ideally, we want the briefest description of these cycles; we want a minimal set of relationships exhibiting interesting cyclic behavior. This minimal set will be a basis for the homology group.

The cyclic relationships in the data are discovered using a boundary map represented as a matrix. To discover the basis we compute the Smith Normal Form of the boundary map. This module computes the homology groups for data represented as an abstract with chain groups \(C_k\) and \(Z_2\). The boundary matrices are represented as rectangular matrices over \(Z_2\). These matrices are diagonalized and represented

\(^{12}\)HyperNetX, Homology and Smith Normal Form: https://pnnl.github.io/HyperNetX/build/algorithms/algorithms.html#homology-and-smith-normal-form
in Smith Normal Form. The kernel and image bases are computed and the Betti
numbers \([45]\) and homology basis are returned.

As an output of the Smith Normal Form "step", we obtained an array of calculated
homology and then invoked interpretation 1.3, which returned the data as represented
in \(C_k\) associated with the array. Later on, we could visualize obtained interpretations.

Our next section will be dealing with model checking and the different formal
methods that exist based on experimental objectives. It will also discuss the use of
those main principles, concepts, and methods to verify several properties.

## 5.4 Formal Methods and Formal Verification Objectives

Software engineering provides various tools for the development. Formal methods are
based on mathematical and logical principles and are often accompanied by automatic
or human assisted tools, they are utilized to formally specify the requirements from
the system. A partition of formal methods is as follows:

- **Modelling:** getting an abstract away some details of the real system applying
  formal ways.

- **Specification:** the first step of the design with the help of such formalism like
  logic, process algebra, and automata.

- **Testing:** checking the execution of the code.

- **Verification:** proving the software correctness with their formal specification.

In recent years, a lot of progress is seen in both, the performance of the different
techniques and tools in accepting the use of formal methods by the software and
hardware industry expecting further innovative creative ideas. Formal methods are
used for:

- Checking the conformity of software requirements with the specifications;
- Testing the process quality to produce;
- Issuing products and quality assurance, defects in the application or the product
and quality assurance;

- The verification aims us to solve some properties:
  - **Connectivity**: determines the process interaction and which are satisfied with nonfunctional requirements;
  - **Correctness**: the system satisfy to either;
  - **Safety**: nothing "bad" will happen, ever, during the execution of a system neither;
  - **Liveness**: "something good" will happen, eventually, when the system executed;
  - **Compensation**: how many services can process models implement;
  - **Compatibility**: bridge the gap between BP design and implementation of enterprise IS and satisfy compatibility between business participants in the collaboration.

## 5.5 Model Checking

Model checking is one of verification approaches which gives system model with a specification of interest to including ensuring basic correctness of processes, business compliance check variability. It looks like an automated error test or simulation; the manner is useful for detecting any property infringement (i.e; bugs or errors) [8].

This formal method used for checking finite-state systems, gives a system model and such specification as a set of formal properties. The model checker verifies whether the model meets the specification or not, and ability its to generate the counterexample [36] (see 5.10) when the model falsifies the specification (*the major advantage of model checker*).

The formal verification of a system needs two inputs: one is the given system description and other, the corresponding properties will be verified.
5.5.1 The First Input: The Systems modelling

Generally, the IS describes modules of information processing systems, links among components, design, and analysis principles at the IS level of which the main purpose is to buttress BP. The representation of systems characteristics as e.g. states, configurations through algebraic methods and techniques can be transcribed into graph representations ([87]) FSM, Kripke structures, Binary Decision Diagrams (BDD). The model extraction from the code can be presented by:

1. **FSM**: a standard model used in the mathematical foundation. It can be conceived as an abstract machine having finite set of states. As there is only one state at the beginning, the change from the current state to another made with transitions is triggered by some event or condition. (All FSM details explained before 3.1.2).

2. **Kripke Structure**: a graph with labeled state transition that can capture the temporal behavior of reactive systems. We can say that the Kripke structure is just a labeled FSM extended to incorporate all labeling function, more formally ([36]) a set of states $s$, transitions and labels for each states defining property. The structure defined as a 4-tuple $M = (S, S_0, R, L)$ where:
   - $S$: the finite set of states;
   - $S_0$: initial states;
   - $R$: the transition relation;
   - $L$: is a labeling function that labels every state with the set of atomic propositions that are true in that state.

3. **Binary Decision Diagrams (BDD)**: a quite old efficient technique to repre-
senting state transition systems, it works as the algorithmic basis for symbolic model checkers.

4. **Model Extraction from Code**: checking of code or the implementation code not the model through some automated model extraction approach.

## 5.5.2 The Second Input: Formal Specification

There are three types of formal specification techniques:

1. **Language-based techniques**: a mathematical technique based on predicate calculus approach.

2. **FSM-based techniques**: an extension of programming language to incorporate the representation of SM and rules. Techniques like extended FSM, petri-nets and abstract SM.

3. **Temporal Logical Techniques**: a statement of the ordering of events and their actions. There are two important sub-types: Linear Temporal Logic (LTL) and Computation tree logic (CTL).

   (a) **Linear Temporal Logic (LTL)**: a common specification formalism for formal method tools that is frequently used for specifying properties of reactive and concurrent systems. It describes the allowed executions using temporal operators represented by the syntax: \( \varphi ::= true \mid false \mid p \mid \neg \varphi \mid (\varphi \lor \varphi) \mid (\varphi \land \varphi) \mid (\varphi \rightarrow \varphi) \mid G\varphi \mid R\varphi \mid F\varphi \mid (\varphi W\varphi) \).

   Where: \( p \in AP \), a set of atomic propositions. The propositional variables or letters can represent fixed properties of the states of the program and the various temporal operators syntax are: \( [ \] \) for G (globally); \( <> \) for F (futur); \( <> \) for X (neXt) and \( U \) for U (strong Until). A property holds in a model if it holds on every path starting from the initial state, LTL is used mostly for applications in software verification.

   (b) **Computation Tree Logic (CTL)**: an example of branching temporal logic has path quantifiers such as: \( A \) (for all paths \( \forall \)) and \( E \) (there exists \( \exists \) a path) named universal and existential quantification, respectively. It allows branching time and quantifiers, CTL is used mostly for applications
in hardware verification.

5.6 Hypergraph Based on FSM approach

FSM and Hypergraph are completely different representations that fit different things. We tried to suggest our own approach of how the problem of mapping between them could be solved. In this part, we will describe a hypergraph to model all types of elements and patterns presented by FSM.

First, we will present FSM elements with hypergraph elements, the hypergraph is used to model diverse types of objects or elements (states transitions). We suppose that each kind of vertices corresponds to a state, and each kind of hyperedge corresponds to the relation between them, hyperedge can be labeled depending on transitions in FSM.

Furthermore, various FSM patterns represented by hyperedge can be divided into sub-sets of presenting a special pattern. Later, we will explain the different concepts in detail and describe theoretically how and what are the rules of mapping.

Part one:

As we have outlined previously, the hypergraph should create a flexible structure that expresses diverse types of sub-FSM. This hypergraph can be represented as nodes embedded into hyperedges and hyperarcs along with their interrelationship to describe the roles that are played by nodes and hyperedges. The direction of the hyperarc shows the direction of transition where input and output are vertices (head and tail of hyperarc) which present states. The definition given [67] above explains:

5.7. Definition. The FSM’s elements are represented by hyperedge $h_i$ where $h_i \in E_{FSM}$ set of hyperedges. The elements of $h_i$; are:

- A finite set of $S$ states are represented by $n$ vertices $V_s = \{v_1; v_2, v_3, \ldots, v_n\}, V_s \subseteq S$.
- A finite set of transitions $T$ can be described by hyperarcs belonging to $A_T$ where $h_i \in A_T, A_T \subseteq T = \{h_1; h_2, v_3, \ldots, h_m\}$. The variables, place-holders or attributes can be denoted as a label of a transition that is described by variables that belong to
an attribute type $A_{ttc} = \{T_1; T_2, T_3, \ldots, T_J\}$ the set of types $h_i \in V_a \cup V_{inter} \cup V_f$ where $a, b, c, d \in N$.

**NB:** Vertices can be labelled with integer by $n \in N$ or denominated with arbitrary strings that can be represented by a binary string $s \in \{0, 1\}$. (variable of domain).

After we mapped places and transitions onto concepts and structures of a hypergraph, we will present its various sub-sets:

The set of vertices $V \subseteq S$ be divided into three sub-sets of vertices depending on FSM states: $V_i$, $V_{inter}$, $V_f$ where: $V_i$ describes the initial state, $V_{inter}$ the intermediate states and $V_f$ denotes the final(s) state(s) of the FSM.

The set of arcs (directed edges) $A$ is partitioned into two subsets: $A_{sim}$ and $A_{cond}$ where: $h_i \in A_{cond} \cup A_{cond}$ where:

- $A_{sim}$: represents a simple transition (labelled without condition), $h_i \in A_{sim}$ can be used between vertices: $A_{sim} \subseteq V_{tail} \times V_{head}$ where:

$$V_{tail} = \left\{ \begin{array}{l} \{v_k \mid v_k \in V_i\} \quad \text{XOR} \\ \{v_k \mid v_k \in V_{inter}\} \end{array} \right; \quad V_{head} = \left\{ \begin{array}{l} \{v_k \mid v_k \in V_{inter}\} \quad \text{XOR} \\ \{v_k \mid v_k \in V_f\} \end{array} \right.$$

- $A_{cond}$: represents a transition labeled with condition used between vertices where: $A_{cond} \subseteq V_{tail} \times V_{head}$ where:

$$V_{tail} = \left\{ \begin{array}{l} \{v_k \mid v_k \in V_i\} \quad \text{XOR} \\ \{v_k \mid v_k \in V_{inter}\} \quad \text{XOR} \\ \{v_k \mid v_k \in V_f\} \end{array} \right; \quad V_{head} = \left\{ \begin{array}{l} \{v_k \mid v_k \in V_{inter}\} \quad \text{XOR} \\ \{v_k \mid v_k \in V_f\} \end{array} \right.$$

**NB:** The last two equations will be not true only if:

$$V_f \times V_f \cup A_{cond} = \phi; \quad V_f \times V_f \cup A_{sim} = \phi.$$

**5.8. Definition.** Labelled Hypergraph is a generalized hypergraph that can be extended by some functions and operations: $H = (V, E, label)$ where: $|V| < \infty,$
$E \subseteq V \times V, \mid V \mid < \infty$; i.e.$E$ finite, $\Sigma = \{\sigma \mid \sigma \in \{0,1\}^*\}$ set of labels, as binary strings.

- **label**: $V \rightarrow \Sigma$; is a function that assigns a label by function $\text{label}(v)$ to vertex $v$ so that $L_{node} \subseteq \Sigma; L_{edge} \subseteq \Sigma$.

- **label\text{\_node}**: $V \rightarrow L_{node}$: where $L$ is a set of labels, it is a vertex labeling function.

- **label\text{\_edge}**: $V \rightarrow L_{node}$: where $L$ is a set of labels, it is an edge labeling function.

- **source\text{\_E}**: $E \rightarrow V$: functions return the source vertex of an edge.

- **traget\text{\_E}**: $E \rightarrow V$: function return the source and target vertices of an edge $E$.

- **att\text{\_Att}**: $E \rightarrow V$: Attribute assignment function.

- **source\text{\_Att}**: $Att \rightarrow L_{node}$: the vertex that owns the attribute is returned.

- **traget\text{\_Att}**: $Att \rightarrow D$: data values of attributes are yielded, $D$ the set of data.

The set of data values $D$ can be grasped (efficiency of the representation is left out of the investigation) again as vertices within the hypergraph and it can be interpreted as variables. Over $D$ the values of sets of variables, sets of operations ($OP$) can be used to describe constraints and rules within formulas.

**5.9. Definition.** Limb is either head or tail of hyperarc and designated by $e_{\downarrow}^i$. The set of condition represented by $A_{\text{cond}}$ partitioned into two sets of its edges and vertices by the values of attributes as they belong to either $T_{\text{bool}}$ or $T_{\text{vdom}}$ respectively: $Att \supseteq (Attr_{\text{bool}} \cup Attr_{\text{vdom}}); \{Attr_{\text{bool}} \cap Attr_{\text{vdom}}\} = \emptyset$.

- **$T_{\text{bool}}$**: can take tree logical value as variables (Boolean): $true$ $false$ or null where: $h_{\text{bool}} \in (A_{\text{cond}} \subseteq V_{\text{tail}} \times V_{\text{cond}})$ and $attr(h_{\text{Bool}}) \in T_{\text{Bool}} = \{\text{true, false, null}\}$.

- **$T_{\text{vdom}}$**: represents the type that a variable $v$ can take up as value; i.e. the variable attribute can be valuated where by $\forall v$ variable $\text{Dom}(v) \subseteq T_{\text{vdom}}, T_{\text{vdom}}$ can be defined as string type $T_{\text{vdom}} = \{s \in \{0,1\}^*\}$ of binary strings; or $X = \{i \leq n \in N\}$ a finite alphabet and $T_{\text{vdom}} = \{s \in \{X_i\}_1^n\}$

**NB:** The relation between the two transitions are:

$$A_{\text{sim}} \cup A_{\text{sim}} = A \text{ and } A_{\text{sim}} \cap A_{\text{sim}} = \emptyset.$$
Part Two:

In this part, we mapped FSM patterns presented before in hyperedges:

5.10. Definition. Each FSM's pattern can be represented by hyperedge $h_i \in E_i$ where $E_i$ the set of hyperedges, $E$ can be divided into sub-sets to present patterns:

- $E_{seq}$: present sequence pattern which can use all type of vertices (the tail of the first hyperarc). It can start from any $v_k \in V_i$ if the sequence starts from the first vertex in the ordering relations, or $v_k \in V_{n \text{ter}}$ if not, but cannot start from any $v_k \in V_f$. It can be finished (the head of the last hyperarc) with any vertex $v_{ml} \in V_f$, where: $|\{v_1, \ldots, v_p\}| \leq n, n \in N$ (typically $n < 3$). The directed hyperarc used to express the relationship between vertices $V_i, V_{n \text{ter}}, V_f$, uses only simple hyperarcs $h_{seq1}, h_{seq2}, \ldots, h_{seqp} \in A_{sim} \subseteq A_{seq}$ between vertices because it is a simple structure. The sequence vertices order is expressed with pair (tail and head of hyperarc), where the head of the first will be the tail of the next, till the sequence reaches the last head where: $h_i^+ \cap h_i^- = v_{ij}$ where $v_{ij} \in (h_i, i), v_{ij} \in (h_i, j)$.

The function $\text{label}$ designates the order of vertices belonging to an instance of FSM through numbering and labelling the hyperarcs that connect them, and being the element in a $h_{seq} \in E_{seq}$ if the sequence is complex one, it can be represented with another hyperedge $h_m \in E_{seq}$ if that vertices is not $V_f$ or no cycle from $V_f$.

- $E_{ps}$: describe parallel split pattern, it can represent a parallel vertices included in the same hyperedge: $E_{par} \in E_{ps}$, various parallel vertices $\{v_1, v_2, \ldots, v_n\} \subseteq E_{ps}$ where $V_j \neq V_i$ if $i \neq j \in N$ i.e. only $V_{n \text{ter}}and V_f$ with the same tail from the main vertex of different hyperarcs $h_{ps} \in E_{ps}$ without a connection between them; where:

\[
\forall k_i, k_i \neq l, h_{kl} \cap h_{kl} = \phi \text{ and } h_{ps} \cap h_{kl} = v_{pc}, \forall k_i \in N, k_1, k_2, \ldots, v_{start_{ps}} \in \overrightarrow{h}_{ps}
\]

This structure uses only simple hyperarcs connected with another type of hyperedges which exist if those vertices are not $V_f$ or no cycle from $V_f$.

- $E_{ec}$: represent exclusive choice pattern, a hyperedge $h_{ec} \in E_{ec}$ contain all vertices $\{v_1, v_2, \ldots, v_n\} \subseteq E_{ec}$ i.e. it can contains only two $V_{n \text{ter}}$ and $V_f$ there
are no arcs or hyperedges embedded into any arbitrary $h_{ec-i} \in E_{ec}$ it has the same tail from the main vertex of different hyperarcs with no connection because it is a simple structure where:

$$\forall k_i, k_l, i \neq l, h_{kl} \cap h_{kl} = \phi \text{ and } h_{ec} \cap h_{ki} = v_{ec}, \forall k_i \in N, k_1, k_2, \ldots, v_{start_{ec}} \in \overrightarrow{h_{ec}}$$

To connect the main vertex of the $h_{ec}$ to other vertices we can designate the conditions $A_{cond} \subseteq A_{ec}$ with values that will be assigned to the element of conditions $T_{bool}$ except null (value) because it cannot take up, and attribute values from $T_{vdom}$ which are allowed to describe the available options, to denominate the choice thereby can be connected to another represented structures.

- $h_{mc} \in E_{mc}$: represent the pattern of multiple choice so that all vertices hyperedge contained in $h_{mc}$ represent different states of exclusive choice of FSM, it can contain only two types of vertices $V_{inter}$ and $V_f$ having the same tail from the main vertex of $h_{mc} \in E_{mc}$ without connection because it is a simple structure where:

$$\forall k_i, k_l, i \neq l, h_{kl} \cap h_{kl} = \phi \text{ and } h_{mc} \cap h_{ki} = v_{mc}, \forall k_i \in N, k_1, k_2, \ldots, v_{start_{mc}} \in \overrightarrow{h_{mc}}$$

To connect the main vertex and $h_{mc}$ use only $h_{mc} \in A_{cond}$ to express conditions $A_{cond} \subseteq A_{ec}$ attributes can be used: $T_{bool}$ includ null (value) (difference between $E_{mc}$ and $E_{ec}$) moreover attribute values describe available options, thereby $h_{mc} \in E_{mc}$ can be connected to other structures represented by hyperedge considering that these vertices belong neither $v \in V_f$ nor a cycle from $v_{cycle} \in V_f$

- $E_{mm}$: present the multiple merge, this pattern is more complicated compared to others because multiple states merging vertices that are included in the same hyperedge, it can contains all type of vertices exist $V_1, V_2, \ldots, V_n \subseteq E_{pc}$ which should have the same head from different vertices merged $h_{mm} \in E_{mm}$ to the main vertex which can be only $V_{inter} V_f$ and there is no connection merged vertices; where:

$$\forall k_i, k_l, i \neq l, h_{jk} \cap h_{lj} = \phi \text{ and } h_{mm} \cap h_{jk} = \phi, \exists k_i \in N, k_1, k_2, \ldots$$

This structure uses only the simple hyperarc to $A_{sim} \subseteq A_{mm}$ connect $E_{mm}$ and the main vertex (the main tail). That main vertex can be connected too with another type of hyperedges exist if it is not $V_f$ or no cycle from it.
• \( E_c \): present cycle pattern which is a simple structure presented only with only a hyperarc \( h_{ec} \in E_c \) from hyperedge which presented the head \( E \) or vertex \( V \) to the tail which can be the same as the head if it is presented with a self-transition in FSM will be described the other case that the tail can be another hyperedge \( E \) or vertex \( V \).

We described the basic concepts and the various essential patterns of the representation proposed, the suggested method takes advantages of the basic properties of generalized hypergraphs. The next section presents the problem posed and the verification method used.

5.7 Problem Posed and verification of Model Properties

Design of IS is becoming more complicated because of design errors which can be resulted from interleaved access over shared data, process synchronization, specifications dynamic changes and very likely the misunderstanding and misinterpretation by programmers on business logical specifications, These are reasons why researchers aim to find solutions (approaches and tools) 1.3 and investigate general verification techniques for quality design. Our main goal is to detect the correctness of the hypergraph based on FSM and check its satisfaction with several properties.

A formal method for model verification based on hypergraph mapped in FSM used model checking, we will use one of the verification techniques to define and verify whether some base hypergraph properties satisfy the model or not? We will apply a verification approach to the hypergraph model because it can be an appropriate example of such structure to help in decision.

The verification must formalize the process model mathematically and verification criteria should be well defined because the hypergraph model wants to be verified within the FSM patterns (represented with its semantic), so the model checking is the suitable technique to verify its correctness.

Graphs and hypergraphs are important examples of such structures, our model which is the input based on some concepts are formulated on the base of the Kripke
structure which is as we mentioned before that are a labeled FSM, so the main concepts of the model will be verified using Kripke structure are:

5.11. Definition. Transition System (Kripke structure), with \( (v_0) \) explicit on hypergraph based of FSM. A transition system: \( M = (V, v_0, \rightarrow, L) \) consists of:

- \( V \) finite set of different kind of vertices \( (V_i, V_{inter}); (V_i) \);
- \( v_0 \subseteq V \) the initial vertex \( (v_i) \);
- \( \rightarrow \subseteq v \times v \) transitions between vertices \( (v_i), (v_{inter}) \) and \( (v_f) \), transition (various edges of the model) can be labeled with an event (actions cited before) a guard and a set of effects that represent actions triggering other effects;
- \( V \rightarrow P \) labelling function such that \( \forall v_1 \in V, \exists v_2, S_1 \rightarrow v_2 \subseteq V \times V \).

5.12. Definition. A path \( \pi \) in our system \( M = (V, v_0, \rightarrow, L) \) representing one possible execution of the system that it models, is a (finite or infinite) sequence of vertices and transitions \( \pi = v_0 a_0 v_1 a_1 s_1, \ldots \), where \( v_i \in V, A_i \in A(s_i) \) and \( \delta(v_i, A_i)(v_{i+1}) > 0 \) for all \( i \in \mathbb{N} \).

Such a program given as code hardware description language corresponds to an FSM i.e; a directed graph consisting of nodes (or vertices) and edges. nodes are system states and edges describe transitions, while the propositions give a property at the execution point. Errors in such systems or process can have consequences, hence the urgent need to be able to ensure and guarantee their correctness. Once the model is complete, properties can be required, and verified. Applying to check have focused on finite-state models and either \(\text{CTL} \) or \(\text{LTL} \). It is necessary to specify properties on the finite-state abstracted models.

Formally, the problem can be stated as follows: given a desired property, expressed as a temporal logic formula \( \varphi \), and a structure \( M \) with initial vertex \( v_0 \) decide if \( M \models \varphi \). The concept LTL model checking seeks to answer the question (with starting state omitted): Does \( M \models \varphi \) hold? or, equivalently: Does \( \forall \pi \in \text{Paths}(M). \pi \models 0\varphi \) hold? where(recall) \( \pi \models^{i} \varphi \) means path at position \( i \) satisfies formula \( \varphi \).

In this part, we will give some general properties \([66]\) which can be verified in our model expressed in LTL formulas and introduce some general idea of the model or special property that part of the model (pattern) to be sure about its
correctness. Propositional variables should be known before to express the set of atomic propositions e.g.: $i, f, m$. The syntax of the atomic descriptions of a system in LTL is expressed by using properties of LTL.

- **Property 1**: starting vertex $i$ must respond to final vertex $f$. LTL entails with: $G(S \Rightarrow Ff)$ this property generally expresses that every process starts with the initial state or initial vertex ($v_i$) must respond to one final states or final vertex ($v_f$), the $G$ express for always (globally) and the $F$ for eventually (in the future). This formula ensures that if a process starts then it should finish, i.e.; the system is working.

- **Property 2**: When an initial action occurs, it will eventually be a final action. When intermediate action occurs, it will eventually be in final action LTL expressed with: $G(i \Rightarrow Ff)$ that means globally when the process starts from the first vertex it will be at final action (one of the final actions) in the future; i.e.; this property generally expresses that every process starts with the initial state or initial vertex ($v_i$) must respond to one of the final states or final vertex ($v_f$), the $G$ express $G$ for always (globally.) and the $F$ for eventually (in the future). This formula ensures that if a process starts it should finish; i.e.; the system is working.

- **Property 3**: initial vertex precedes intermediate vertices (if exists) after one of the final vertices. LTL expressed with: $\neg q \cup (q \land \neg p \cup s)$, this property expresses that globally the state expressed by $q$ ($v_f$ final state) could not happen only until the state $p v_{inter}$ intermediate state (if it exists in the process) happen and this state too can happen if only the initial state starts the process ($v_i$), this property can express the respecting of the order in the model to guarantee the good sequence of different actions.

- **Property 4**: $S$ exists between $Q$ and $R$. LTL expressed with: $G(Q \land \neg R \Rightarrow \neg RW(S \land \neg R))$, this property generally expresses that every intermediate state $R$ in LTL expression should be happening between initial expressed with $Q$ state or initial vertex ($v_i$) and one of the final states $R$ or final vertices ($v_f$) the $G$ express $G$ for always (globally) and the $W$ for weak Until.

This formula ensures the order of the vertices or states in the model should be respected i.e.; an intermediate ($v_{inter}$) state should be between the initial state
(vᵢ) and final one(vᵢ). So, the system is working in respected order.

- **Property 5**: Checks that the model is non-trivial. LTL expressed is: $\text{Extrue}$, this property generally expresses that at least one state success, $G$ for always (globally) and the $F$ for eventually (in the future). This formula ensures that the model is significant.

Generally, two types of properties can be expressed using temporal logic: Safety and Liveness. Safety property states that something bad never happens, a simple example of that is the LTL formula $G\neg\text{error}$ that means that error never occurs.

**Conclusion**

In this section, we presented the concept of hypergraph, different terms and definitions used are defined. We tried to represent its implementation as we described, after that, we aligned the basics of the hypergraph representation based on FSM. Several patterns and equations were explained in detail, the last part presented different formal verification tools and their importance, some concepts of formal methods were used to verify the correctness of the hypergraph based on FSM properties.
Chapter 6

Conclusion

6.1 Contribution

In this dissertation (described in section 1.3), we addressed various problems related to the BP concept. Our contribution to this dissertation is explained here:

- In order to capture essential concepts of BP models and represent them in a more understandable and expletive manner. We made a comparison represented in table 3.2 and table 3.3 between existing BP models and we established the existing differences and the similarities. We found that there are similar syntax and functionalities to make transformations between these models;

- The comparison between BP models (in section 3.4) also helped to create combinations and integrate between them for a formal verification that was not possible before;

- We focused more on BP functionalities i.e dynamic aspect (in chapter 4) to conceptualize and enrich the domain, unlike the usual focus on static aspects;

- The various changes which affect the BP were defined in details, items which could be affected were presented and their functionalities which could be broken, stopped, or modified were more specified for future use betterment and impact minimization;

- BP modelling using various models gives this BP the capacity for easier
management. In our studies, we tried to represent the concept of a hypergraph and implement some representations and matrices and forms that can generally help to use many tools for verification of the process and check some of its properties by combining with other models like FSM.

6.2 Conclusion and Future Work

The work in this dissertation represented the investigation on BP modelling and its management reached that its completion, we tried to answer the questions we say in chapter 1.

We began our dissertation with the theoretical background summarizing our research domain and definitions. We started with the notions of enterprise, IS and its components. The main focus was on the concept of BP, and its modelling. Many BP technologies were described as well like: workflow, web services, EA . . .

We thoroughly reviewed the literature written about existing BP models such as Petri-nets, FSM, BPMN, UML diagrams . . ., we based on semantic, syntactic, and structural criteria of each model to insensitively compare between them, to point out the main differences and to conclude negative and positive points for each, this also helped in concluding methods of transformations between those BP and languages. A set of transformations done during our studies were presented well in this chapter.

In chapter 4, we aimed to report on a study that looked thoroughly at the dynamic aspect of BP, and we further discussed the changes that impact the process during its functioning time and their adjustment. We concluded with our definition of dynamic BP conceptualization. We also answered most of the questions related to dynamic BP posed before.

In the last chapter of our dissertation, the concept of the hypergraph, its representation, and the approach of the hypergraph based on the FSM approach was represented. Formal verification will be applied to verify several properties.

We concentrated on the concept of the hypergraph and its representation and because it was implemented in different approaches and forms, this helped us in our future work to use it directly to represent process problems and for more complex
types of BP such as health processes, which are characterized by being sensitive to
the contexts in which they are implemented and are therefore highly variable. As well
as economical BP which is more influenced by environments (internal and external)
and the current state of coronavirus, this will be our future work.

I am aiming to conduct further research that goes under applying of process
mining and machine learning approaches (the proposal idea discussed) in order to
verify and align BP (focusing on health and economic BP that are sensitive and
more impacted in last the world state) using BPMN model.

As other future work will be the implementation of the published study [18]
consisting of an application to represent the EA (actual research) and verify it using
the representation [82] and different forms to demonstrate its use and efficiency.
Bibliography


[77] Frank Leymann. Workflow-based coordination and cooperation in a service world. In OTM Confederated International Conferences" On the Move to


[141] JianHong Ye, ShiXin Sun, Lijie Wen, and Wen Song. Transformation of bpmn to yawl. In *2008 International Conference on Computer Science and Software*


