UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL
INSTITUTO DE INFORMÁTICA
CURSO DE CIÊNCIA DA COMPUTAÇÃO

NICOLAS MAURO DE MOREIRA BOHNENBERGER

Healthcare Process Discovery from Spread Out Natural Language Documentation: a SARS-CoV-2 and COVID-19 case study

Work presented in partial fulfillment
of the requirements for the degree of
Bachelor in Computer Science

Advisor: Prof. Dr. Lucineia Heloisa Thom
Coadvisor: Alessandra Ceolin Schmitt, M.D.

Porto Alegre
May 2021

ABSTRACT

The discovery of business process information in natural language documents is challenging, as natural language can be ambiguous and unclear at times, and can become even more challenging with information on such processes displayed in different ways throughout their documentation, and spread out between different documents. This issue is even more critical when one is faced with complex and dynamic process domains, such as healthcare. This work presents an approach to the discovery of healthcare processes from spread out, natural language documentation, that generates process models and their text descriptions, which comprises the search for documentation, data extraction from it, process model generation from natural language text, and finally text description extraction from the process models. This approach is then applied to a case study on SARS-CoV-2 and COVID-19, which results in process models and descriptions for SARS-CoV-2 Transmission and Contagion, and COVID-19 Symptomatic Manifestation and Identification processes. These process models and descriptions are then semantically validated by a domain expert and evaluated by business process modeling experts. The process models are also structurally verified with the use of Petri-net-based analysis. The main contribution of this work is an approach to the discovery of healthcare processes from spread out documentation. The results of this work are a validated and verified set of healthcare process models comprised of the SARS-CoV-2 Transmission and SARS-CoV-2 Contagion processes and of the COVID-19 Symptomatic Manifestation and COVID-19 Identification processes.

Keywords: Business Process Management. Business Process Discovery. SARS-CoV-2. COVID-19.

Descoberta de Processos da Saúde a partir de Documentação Dispersa em Linguagem Natural: um caso de estudo sobre SARS-CoV-2 e COVID-19

RESUMO

A descoberta de informações de processos de negócio em documentos de linguagem natural é desafiadora, dado que a linguagem natural pode ser por vezes ambígua, e pode se tornar ainda mais desafiadora com as informações sobre estes processos apresentadas de diferentes formas por sua documentação, e espalhada entre documentos de diferentes tipos. Este problema se torna ainda mais crítico quando nos deparamos com domínios de processo complexos, como cuidados da saúde. Esta monografia apresenta uma abordagem para a descoberta de processos da saúde a partir documentação dispersa e em linguagem natural, composta por busca por documentação, extração de dados, geração de modelos de processo a partir de linguagem natural, e extração de descrição textual de processos a partir de modelos de processos. Esta abordagem é então aplicada a um estudo de caso sobre SARS-CoV-2 e COVID-19, que resulta em modelos de processo e descrições de processo para a Transmissão e Contágio do SARS-CoV-2, e Manifestação Simtomática e Identificação da COVID-19. Estes modelos de processo e descrições são então validados semanticamente por uma especialista de domínio e avaliados estruturalmente por especialistas em modelagem de processos de negócio. Os modelos de processo também são verificados estruturalmente por meio de análises baseadas em Redes de Petri. A principal contribuição desta monografia é uma abordagem para a descoberta de processos da saúde a partir de documentação dispersa. O resultado deste trabalho é um conjunto validado e verificado de modelos de processos da saúde composto pelos processos de Transmissão do SARS-CoV-2, Contágio do SARS-CoV-2, Manifestação Sintomática da COVID-19, e Identificação da COVID-19.

Palavras-chave: Gerenciamento de Processos de Negócio. Descoberta de Processos. SARS-CoV-2. COVID-19.

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS AND ACRONYMS

7PMG	7 Process Modeling Guidelines
BPM	Business Process Management
BPMN	Business Process Model and Notation 2.0
PN	Petri Net

CONTENTS

1 INTRODUCTION

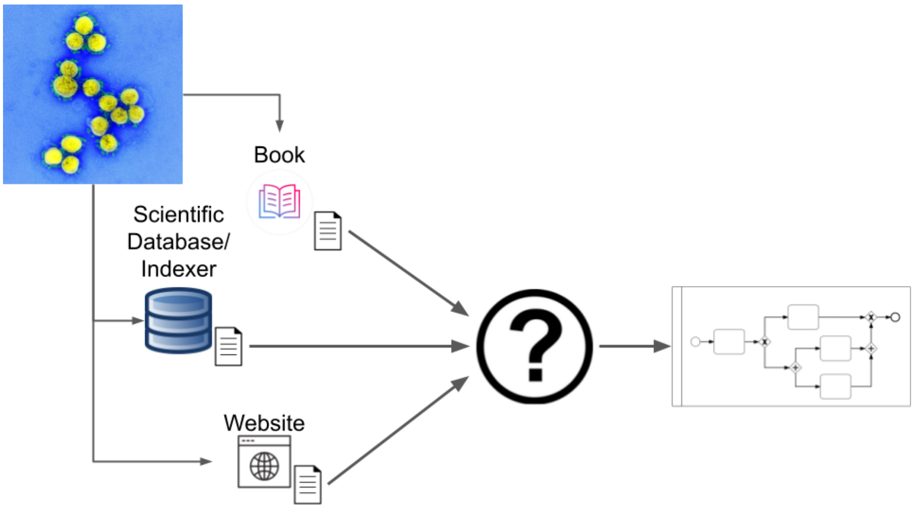
A business process is a set of activities, decisions and events that, when performed in coordination, collectively deliver an outcome that brings value to an organization (DUMAS et al., 2018). It is performed by a single organization, by multiple participants, but it may interact with other processes performed by other organizations (DUMAS et al., 2018; WESKE, 2012). In the discipline of Business Process Management - BPM, Process Discovery is the phase of the BPM Lifecycle in which information on a process is collected and organized, and then modeled into what are called as-is process models, reflecting what is understood about how that process works in visual form (DUMAS et al., 2018). Designing the as-is process model is not the sole activity in process discovery, but a part of that phase of the lifecycle; gathering the information needed usually proves to be more consuming (DUMAS et al., 2018). One of the methods through which Process Discovery can be performed is document analysis, consisting of extracting relevant information on processes from documentation available related to them (DUMAS et al., 2018; WESKE, 2012).

However, documentation can take many forms (internal policies, organization charts, quality certificate reports, handbooks, etc.) and lack standardization, which can make extracting process information from its documents a complex task. Outside of an organization environment, in which documentation might be easier to find and gather, collecting documents to analyze is not straightforward. Dynamic process domains, such as healthcare, can make this task even more challenging, due to their particular characteristics (THOM et al., 2010; REICHERT, 2011). In this domain, much of the documentation is spread out, taking the form of different scientific papers, technical reports, and books, each with their own approach to the same object of discovery. They are also commonly written in natural language, which can be ambiguous and unclear. Moreover, few of these documents seek to provide a visual or behavioral perspective to their reported processes, usually focusing on a specific specialty within the healthcare domain.

A recent subject in the healthcare process domain is the COVID-19 pandemic. The SARS-CoV-2 virus is its known causative agent. The disease was first discovered in December 2019, in the Hubei province of China (ProMed, 2020), and quickly spread to various countries, putting the world in a state of alert. As of March 2021, the World Health Organization - WHO reported over 118 million confirmed cases and over 2.5 million confirmed deaths caused by COVID-19 worldwide (WORLD HEALTH ORGANI-

ZATION, 2021). The pandemic was met with swift response from the scientific community, generating knowledge that allowed for diagnostic testing, epidemiology tracking, and development of therapeutic strategies. Moreover, a variety of case series and reports on clinical findings have been published detailing the various effects of the viral infection in different regions of the world (WANG et al., 2020a; DU et al., 2020). Documentation on these findings and numerous others, as well as literature reviews with the intent of aiding comprehension of the disease have been gathered in multiple repositories. WHO maintains a COVID-19 indexing database updated regularly from bibliographic databases and scientific articles, as well as a list of resources on coronavirus infections such as the British Medical Journal - BMJ, Elsevier, the Journal of the American Medical Association - JAMA Network, The Lancet, Springer Nature, and others on their web page (WORLD HEALTH ORGANIZATION, 2020a). Due to this, to design SARS-CoV-2 and COVID-19 processes, there is a need to collect documentation from all of these sources, identify the presence of information pertinent to process design in said documentation, and finally combine this information into a single process description. An illustration of this issue is presented in Figure 1.1.

Figure 1.1: Illustration of the issue of modeling SARS-CoV-2 and COVID-19 processes from spread out documentation.



Source: The authors. (Transmission electron micrograph of SARS-CoV-2 particles by NIAID (2021))

1.1 Goals and Hypotheses

This work has the following hypotheses: H1) Through a process-oriented approach, it is possible to consolidate spread out documentation for the discovery of process information and process design; and H2) The Business Process Model and Notation 2.0 -

BPMN 2.0 (OMG, 2014) is capable of representing behavioral information of the SARS-CoV-2 virus and the COVID-19 infection.

The main goals of this work are to present an approach for the discovery of healthcare processes from spread out, natural language documentation, with the purpose of generating visual and behavioral documentation of these processes in the form of BPMN 2.0 process models and their text descriptions, and to generate visual and behavioral documentation of SARS-CoV-2 and COVID-19 processes. The approach presented encompasses the search for documentation, data extraction, model generation from natural language and finally text extraction from process models. We demonstrate its application to a case study on the SARS-CoV-2 virus and the COVID-19 infection, and the resulting process models and process text descriptions for the SARS-CoV-2 Transmission, SARS-CoV-2 Contagion, COVID-19 Symptomatic Manifestation, and COVID-19 Identification processes. Afterwards, we validate it semantically with a domain expert and structurally with a BPM expert, and verify the process models structurally through a Petri-net-based analysis Verbeek and Aalst (2000).

1.2 Main Contributions

The main contributions of this work are a proposed approach for discovering processes in spread out documentation, extracting data from this documentation, and modeling them into BPMN 2.0 process models, also generating a structured process text description in natural language, and visual and text documentation on the behaviour of SARS-CoV-2 and COVID-19 processes. Another contribution developed during this work is a conversion script to allow for XML and CPN files describing Petri nets to be converted to TPN files, so that they can be analyzed by the Petri-based-analysis tool Woflan (VERBEEK; AALST, 2000). We also expect this work to encourage the use of BPMN 2.0 notation for the design of complex domain processes.

1.3 Text Organization

This work is organized as follows. Chapter 2 presents the background of this work, including BPM, BPMN, the subject of the case study, SARS-CoV-2 and COVID-19, and related works. Chapter 3 presents the approach developed for discovering processes from spread out documentation, and its application to SARS-CoV-2 and COVID-19 processes, and also the resulting process models and text descriptions. Chapter 4 presents the validation of the results by a domain expert and verification of the models via Petri Nets - PNs. Chapter 5 concludes the work and presents future directions.

2 BACKGROUND AND RELATED WORKS

In this chapter, we present the fundamental concepts related to BPM, a description of the elements of the BPMN 2.0 notation employed throughout this work, background on the subject of our case study, the SARS-CoV-2 virus and COVID-19 infection, and related works.

2.1 Business Process Management

BPM is the discipline of overseeing the performance of work in an organization with the goal of ensuring consistent outcomes and seeking to take advantage of opportunities for improvement in it (DUMAS et al., 2018). It is based on the notion that the combination of activities performed are what compose a product that a company delivers to market (WESKE, 2012).

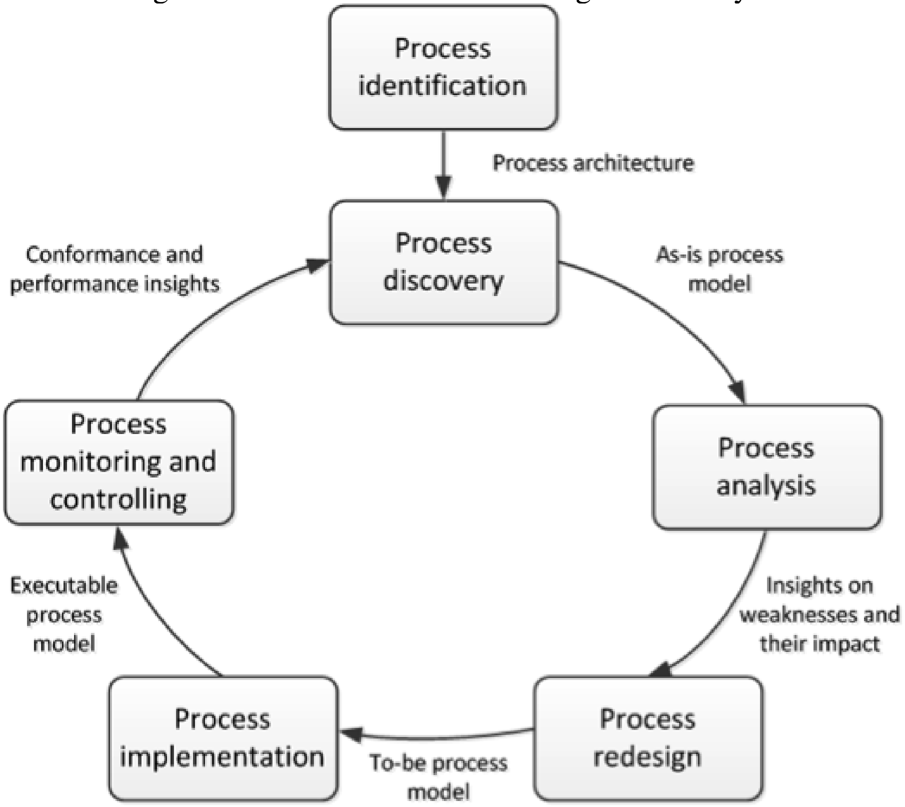
BPM aids in the reduction of costs, of executions times, and of error rates in process executions, resulting in an increase of process quality and in process results (DUMAS et al., 2018). BPM is a discipline that has been applied to different application domains, including dynamic environments such as healthcare, a domain for which processes may need continuous adaptation without the outcome being affected (THOM et al., 2010; REICHERT, 2011).

According to Dumas et al. (2018), the BPM lifecycle (Figure 2.1) has six different phases: *process identification*, *process discovery*, *process analysis*, *process redesign*, *process implementation*, and *process control and monitoring*.

Process Identification aims to define what business processes pertain to an organization or subject, and to establish criteria for selecting processes to improve. The result of this phase of the lifecyle is a *process architecture*, representing processes and their relations to one another, which helps define priorities for modeling and redesign efforts. Processes that receive priority are usually those that are more important, strategically speaking, to the organization, or those that have more problems which when solved generate improvements for all stakeholders.

Process Discovery is the phase of the BPM lifecycle in which information on key processes is inferred and understood to create *as-is process models*: representations of the existing knowledge on relevant processes in visual and standardized form.

Figure 2.1: Business Process Management lifecycle



Source: Dumas et al. (2018)

Process Analysis is the point in the lifecycle where as-is process models are investigated and issues related to them are identified and documented, and when possible quantified. This phase of the lifecycle has as output a collection of issues that are then prioritized according to effort required to resolve them and impact they may have on the overall performance of the processes.

Process Redesign takes the issues identified in process analysis and looks for changes that can be made to the processes as identified in process discovery to address them and help the organization reach their objectives with regards to performance, proposing various options and comparing them with one another. As a result, some of these options are incorporated into a *to-be process model*, a redesigned version of the process being looked at.

Process Implementation is when the *as-is process* is altered and modified into the to-be process. This entails both organizational change management and automation of the process. Organizational change management is related to the changes in the activities required for the new process to be adopted by all participants. Automation of the process involves developing and deploying IT systems, or enhancements of existing systems, that will support the to-be process.

Process Monitoring occurs after the implementation of the process. In this phase, data on process execution is collected and then analyzed to determine how well the process is performing compared to different metrics and established objectives. Often in this phase bottlenecks and recurrent errors in execution are identified and corrected.

This work focuses on the first two phases of the BPM lifecycle, Process Identification and Process Discovery.

Process Discovery can be considered as one of the most crucial phases of the BPM lifecycle, as its goals are to make it easier for stakeholders to understand the process as a whole, and to facilitate its conception (PRIEGO-ROCHE et al., 2012). The design of as-is process models, however, can't begin before information on them has been collected, a task that is commonly cumbersome and time-consuming. Moreover, since modeling and domain knowledge are not commonly present in the same person, there are fundamental challenges to be overcome in process discovery (DUMAS et al., 2018):

- Fragmented process knowledge, a byproduct of division and specialization of labor which renders process information scattered among various resources, which in turn have detailed knowledge of their task, but not of the process as a whole. This creates a need for process discovery to require various iterations of discovery activities to reach a consensus or compromise from the domain experts and process owners.
- Domain experts usually see the processes they take part in on a case-by-case basis, being able to describe actions taken for particular instances of their tasks, but not being able to describe generally how that task is executed, or where it stands in the process.
- Domain experts are frequently not acquainted with business process modeling languages, which means not only they are not trained in creating process models, but also in reading and understanding process models in general. This makes getting useful feedback harder from the process models on their own, usually generating the need to also describe the business processes in natural language for easier comprehension.

Several methods have been proposed in the literature for the discovery of business processes towards as-is process model design. Dumas et al. (2018) distinguishes three classes of process discovery methods: evidence-based discovery, interview-based discovery, and workshop-based discovery. Each of these approaches presents their own advantages and disadvantages.

Evidence-Based Discovery methods are those that rely on existing evidence of how processes work, and that can be studied to understand these processes. Among them, document analysis, observation, and automated process discovery are most common.

Document analysis takes advantage of the fact that documents exist that are related or that can be related to business processes, such as internal policies, organization charts, employment plans, reports, handbooks, data models and others, from which knowledge can be extracted and applied to process design. This can give a process analyst an overview of processes, and the ability to formulate hypotheses, prior to talking to domain experts. Document analysis, however, present disadvantages. Most documents are not written or assembled in a process-oriented fashion, which costs time to be translated to a format useful to process design. Another possible issue is that documents might present information on specific tasks in the process that are too detailed to be modeled at the process level. A third possible complication is that documentation might not necessarily represent reality faithfully, being outdated by alterations in the actual, real-life practices caused by any number of factors, or are simply normative documents that describe the ideal circumstances of the process.

Observation methods involve process analysts directly following individual cases to acquire understanding on the workings of processes, either from an active role in the process, or passively observing its execution (DUMAS et al., 2018). When taking active participation in the process, the analyst triggers an instance of the process (as a customer, for example) and records how it progresses and what choices were available through it. This gives them an understanding of the process and which are its more significant milestones, but also has their observation of the process limited by the point of view of their participation in it. When observing the process passively, the analyst has a much wider view of the inner workings of the process, being able to observe it from many angles, but this requires access to locations and resources where the process happens, which might not always be available. Moreover, process actors might behave differently knowing they are being observed, which can give the analyst a representation of the process distorted from reality. Process discovery based on observation has the benefit of presenting how business processes are happening in reality.

Automated process discovery relies on data stored in organizational systems related to the processes being observed, or *event logs*, to discover process models that these systems support. Event logs show the reality of the execution of the business process, giving an objective representation of the process (AALST, 2016). They also often store other information related to processes, such as resources involved in it, time expended on tasks, among others. This information complements performance information that can be discovered in process monitoring, and can aid in improving the process. Moreover, automated discovery is not limited to a single system, but is able to discover processes that are executed across entirely different systems and organize information that, isolated in one particular system, would be much harder to use effectively. However, the availability of event logs and their coverage of processes may have gaps, or contain useless information and logging errors. Additionally, the activities related to the process may be too detailed, in such a way that reading the discovered process proves difficult.

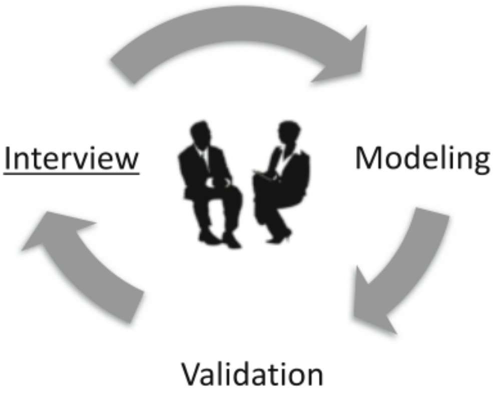
Interview-based Discovery is dependent interviews with domain experts for the design of the process. Since knowledge of the process is usually fragmented between different organization resources, interviews with multiple domain experts are required for a good grasp of the whole business process.

One possible strategy for conducting a discovery interview is starting from the process trigger until the outcome is reached, which has the benefit of following the natural flow of the process. Another possibility is to do the opposite, beginning at the outcome of the process and working backwards towards its trigger, so that requirements for the outcomes of each activity are identified. These strategies are complementary to each other.

Interviews commonly employ structured and free-form approaches. A structured approach can take the form of a list of questions that elicit process details and can be used to validate current hypotheses the analyst has about a business process, but may fail at uncovering information that is not explicitly requested as an answer. A free-form approach allows domain experts to discuss the process and the activities they are more familiar with at the level they choose, and may uncover unknown or disregarded information about the process.

After a first round of interviews is completed, the analyst designs a process model based on recorded information (audio recordings, notes taken, etc.). This process model is then presented to the domain experts for validation, to ensure that it is consistent with their view of the process. It may be necessary, given domain experts not being familiar with modeling languages, to translate the process model to natural language so that it is properly understood. Validation commonly leads to another round of interviews, which initiates another iteration of the approach, as illustrated in Fig. 2.2. Once the process is approved by the interviewees, the business process model is validated.

Figure 2.2: Interview-based Discovery cycle



Source: Dumas et al. (2018)

Workshop-Based Discovery gathers multiple participants at the same time in an effort to achieve a rich understanding of the business process. It usually requires a facilitator to coordinate contributions of all participants, and a process modeler which designs the process in real-time according to the contributions of the participants, besides the business analyst, which takes note of concerns that may need to be further investigated.

Modeling a business process end-to-end in a single workshop session is uncommon, as views from different domain experts may clash and have to be resolved. Preparing and scheduling these workshops in advance is decisive for their success, gathering multiple domain experts who have different involvement in its execution.

A possible technique for the workshop is to ask the participants to sketch a rough model of the process using blank cards or sticky notes, following the flow of an execution of the process, taking care to maintain the same level of granularity or scope from one task to the next. Participants may disagree on the scope or actions involved in a task, which should be handled either by reaching a consensus or creating a composite activity from the different options presented. This exercise leads to a sketch of a process model that can be used as input for the modeling of the business process in BPMN after the workshop session, or during the session if the role of process modeler has been employed in the session.

Workshop-based process discovery relies on organization and facilitation so that all participants feel welcome and willing to discuss the process and participate in the discovery effort. Selection of participants should be considered, and stimulating constructive interaction between participants aids in reaching a consolidated opinion on the process.

2.2 Business Process Model and Notation

BPMN 2.0 is a standard for business process design that provides a graphical notation meant to be understandable by all business users involved in the creation, implementation, management and monitoring of business processes(OMG, 2014). It creates a bridge between process design and implementation. Furthermore, it has the goal of ensuring XML languages designed for execution of business processes can be visualized in a notation that is business oriented (WORKFLOW MANAGEMENT COALITION, 2012). It is an Object Management Group - OMG specification, and is also ratified as an International Standard (ISO/IEC 19510) (INTERNATIONAL ORGANIZATION FOR STANDARDIZATION, 2013). BPMN 2.0 has five element categories named Flow Objects, Swimlanes, Artifacts, Connecting Objects, and Data. In the modeling of the processes contained in this document, only the first four are employed, therefore in this section we will expand on those categories and the elements used in these processes.

Flow Objects are the nodes in a Business Process Model; they are the main elements that define the behavior of a Business Process. There are three types of Flow Objects: activities, events, and gateways.

Activities are tasks an organization performs in a process. Activities are represented as rectangles with rounded corners, and can be atomic or non-atomic. Activities include Tasks (Figure 2.3), such as “Gather patient information”, or “Inhale infected droplets”, which are atomic activities, and sub-processes, which are compound activities included within processes. *Events* are things that happen instantaneously over the course of a process. They affect how the model flows and normally have a cause, or trigger, or an impact, or result, on the process. They are represented as circles with open centers, and internal markers distinguish different triggers or results. Events can indicate where a particular process starts (*start events*) or ends (*end events*), or can occur between the start and end of a process (*intermediate events*).

Figure 2.3: Representation of the Task element

Source: The authors.

Start events (Figure 2.4) define how (and where in the diagram) a particular process starts, and are represented as circles with thin borders, such as "Entered in contact with virus". Start events can be further differentiated as Start message events, such as

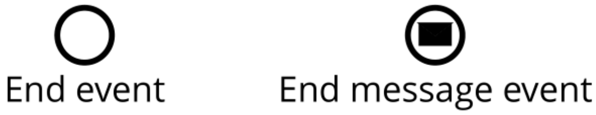
“Patient with difficulty breathing arrived”, among others, to more clearly show what was the trigger for starting the process. A start message event indicates that a process starts upon receipt of a message from another party.

Figure 2.4: Representation of Start event elements

Source: The authors.

End events (Figure 2.5) define how (and where in the diagram) a particular process ends, and are represented as circles with thick borders. End events can be further differentiated as End message events, which are represented by being drawn with an envelope inside them, such as “Professional help sought out”, among others, to more clearly show what was the result of the ending of the process. An end message events indicates the process ends by sending out a message to another party.

Figure 2.5: Representation of End event elements

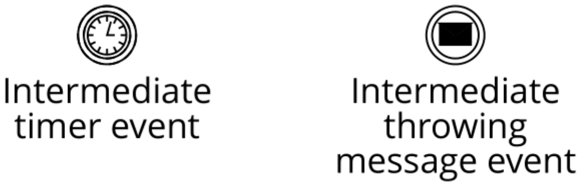


Source: The authors.

Intermediate events (Figure 2.6) can occur only between a Start Event and an End Event, but do not start or directly terminate a process, and are represented as circles with double borders. They can either react to (catch) a trigger, or create (throw) a trigger, and can be defined by what type of trigger they catch or throw, such as intermediate timer events. Intermediate timer events can only catch triggers, since time is beyond the control of the process, and represent a temporal interval that needs to pass before a process instance can proceed beyond it, such as “Incubation period”, or “14 days”. Intermediate timer events are differentiated from other intermediate events by being drawn with a clock inside them.

Intermediate events can also be employed in the handling of exceptions, or “rainy-day scenarios” of a process (DUMAS et al., 2018), such as Intermediate Error Events (Figure 2.7). Intermediate error events are triggered when something internal to the process deviates from the expected process flow, and they are differentiated from other in-

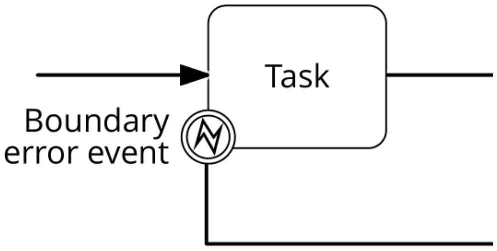
Figure 2.6: Representation of Intermediate event elements



Source: The authors.

termediate events by being drawn with a lightning marker. When intermediate events are used for that purpose, they are attached to the boundary of a task, such as in "Clinical condition worsened". When a boundary event catches a trigger, it triggers the recovery procedure for that exception, the exception flow.

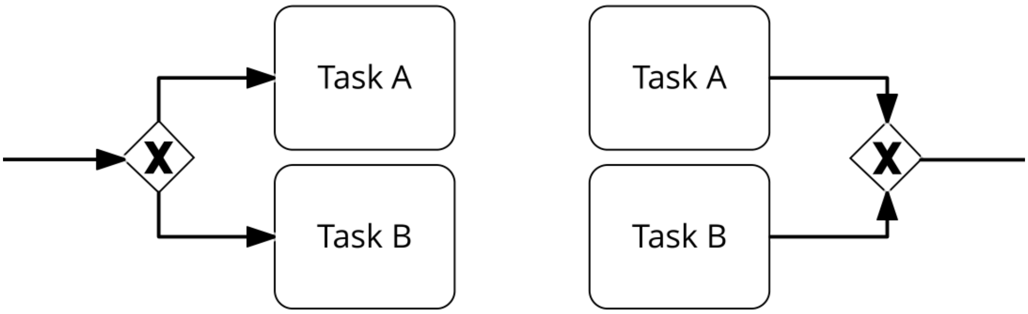
Figure 2.7: Representation of a boundary intermediate event element



Source: The authors.

Gateways are used to determine branching, forking, merging, and joining of paths in a business process. They are represented by a diamond shape, and internal markers distinguish between different types of behavior control. Types of control include exclusive decision and exclusive merging, and parallel forking and joining.

Figure 2.8: Representation of Exclusive Choice and Exclusive Merging gateways

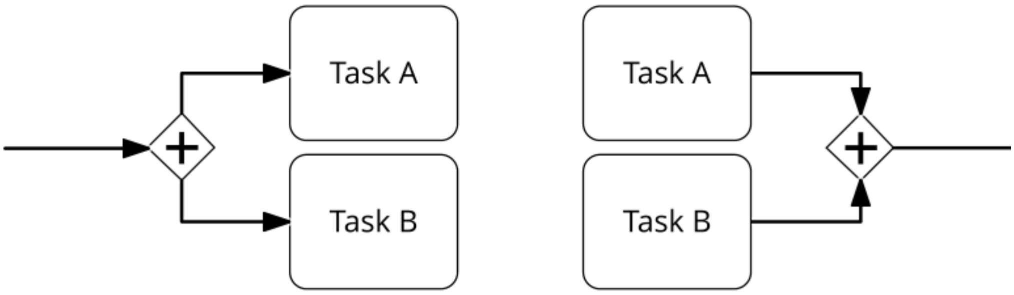


Source: The authors.

Exclusive decision and exclusive merging gateways, also known as XOR gateways (Figure 2.8), such as “Difficulty breathing?”, are gateways in which one choice excludes all other possibilities for that decision point. They are represented by an “X” inside the gateway shape.

Parallel forking and joining gateways, also known as AND gateways (Figure 2.9), as shown in the ’COVID-19 Identification’ process, are employed to create and join (synchronize) parallel flows. They are represented by a plus (“+”) sign inside the gateway shape.

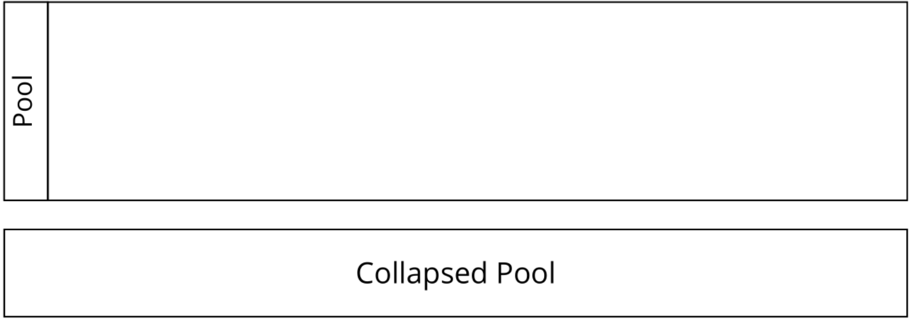
Figure 2.9: Representation of Parallel Forking and Parallel Joining gateways



Source: The authors.

Swimlanes model active resources involved in the process, that is, resources that can perform activities on their own, also called participants (Dumas et al. 2018). Swimlanes can be pools or lanes. In this work, we only make use of Pools (Figure 2.10). Pools are graphical representations of participants in a process, and graphical containers for sets of Activities of that process pertaining to a particular participant. They are represented as rectangles in a process, and may have internal details - the process that will be executed - or not, in which case they are called black box or collapsed pools.

Figure 2.10: Representation of Pool and Collapsed Pool elements



Source: The authors.

Artifacts are used to expand on the information about a process inside the model. There are two standardized Artifacts: groups and text annotations (INTERNATIONAL ORGANIZATION FOR STANDARDIZATION, 2013). In this paper, we make use of text annotations only (Figure 2.11). They are a mechanism for providing additional information through text in a Business Process Diagram, to better inform the reader about the process and its details.

Figure 2.11: Representation of a Text annotation element

Source: The authors.

Finally, *Connecting Objects* are used to link Flow Objects to one another or to other information. They include Sequence Flows, Message Flows, Associations, and Data Associations. In this work, we make use of the first three. Sequence Flows (Figure 2.12) are utilized to show in which order Activities in a process are performed, and are represented by solid arrows between Flow Objects. Message Flows (Figure 2.13) show the flow of messages between participants in a process, and are represented by empty arrows with dashed lines connecting Flow Objects, with an empty dot showing their starting point. Associations (Figure 2.14) are used to connect information and Artifacts to BPMN graphical elements, and are represented by a dashed line.

Figure 2.12: Representation of a Sequence Flow

Source: The authors.

Figure 2.13: Representation of a Message Flow

Source: The authors.

Figure 2.14: Representation of an Association

Source: The authors.

2.3 Process Model Quality

Though BPMN 2.0 notation seeks to create business process models that are understandable by all parties involved in a business process, a business process model will not be of help to those parties if it's not a good process model. The quality of a process model can be evaluated in three aspects: *syntatic*, *semantic*, and *pragmatic* (DUMAS et al., 2018; REIJERS; Mendling; Recker, 2010).

Syntactic Quality is the aspect of quality related to compliance to the rules of the technique or notation a model is designed with (REIJERS; Mendling; Recker, 2010). In the case of BPMN process models, it refers to how the process model conforms to the rules established by BPMN 2.0; in the case of Petri Nets (REISIG, 1985), how a net conforms to the definition of the class of Petri Nets it's supposed to be in.

Syntactic quality can be *verified*, that is, checked for formal properties of the model without the need for any knowledge of the real-world process being modeled (REIJERS; Mendling; Recker, 2010). These properties can be divided between static and behavioral properties. *Static properties* are related to the type of elements that are used in a model, and how they are connected amongst themselves. For example, in a BPMN model, it's not allowed for a message flow to connect two elements within the same swimlane (OMG, 2014). *Behavioral properties* are related to completion of process instances of that model, to which different correctness criteria apply (REIJERS; Mendling; Recker, 2010). Behavioral properties can be checked automatically by software. A prominent example is Aalst (1997)'s *soundness* property, which requires that:

1. In any state of a model, it has the option to complete.
2. Every completion of a model execution leaves no branches still active.
3. There are no tasks in the model that can never be executed.

Semantic Quality is the aspect of quality related to how well does the process model capture the real-world process being modeled (REIJERS; Mendling; Recker, 2010). Semantic quality is tied to the meaning of what is being modeled. It can be decomposed into *validity* and *completeness*: *validity* measures how correct and relevant the statement the model makes are to the process, while *completeness* measures if the model contains all relevant statements that *would* be correct about the real-world process. Semantic quality is relative, relying on less explicit guidelines than syntactic quality (REIJERS; Mendling; Recker, 2010).

Semantic quality can be *validated*; two particular techniques that stand out are *simulation* and *paraphrazation* (REIJERS; Mendling; Recker, 2010). *Simulation* presents the behavior of a model to users in a visual and intuitive way, using animation to help them visualize different paths and decisions that have to be taken to achieve a particular result. *Paraphrazation* seeks to bridge the gap between the user and the modeling language by transporting the model back to a natural language text, which can be understood and discussed more generally than a particular modeling language would.

Pragmatic Quality is the aspect of quality concerned with how understandable and useful is a process model. It differs from semantic quality in the sense that a process that leaves out big portions of the real world, thus having low semantic quality, may be understood perfectly in terms of relations expressed between its elements, indicating a high pragmatic quality (REIJERS; Mendling; Recker, 2010).

Pragmatic quality can be *certified*, however it is usually not approached systematically, becoming more of a sign-off by a process owner with regards to the clarity and readability of the process model. For designing process models, the 7PMG (subsection 2.3) are guidelines to help achieve pragmatic quality. Pragmatic quality is the least understood aspect of process quality, as few scientific works on this aspect have been developed (REIJERS; Mendling; Recker, 2010).

The 7 Process Modeling Guidelines - 7PMG are a set of guidelines for ensuring the pragmatic quality of a model, built on strong empirical insights but formulated to be intuitive to users (MENDLING; REIJERS; AALST, 2010). The guidelines are as follows:

G1: Use as few elements in the model as possible. The size of a model has an impact on its understandability and chance of containing modeling errors. Larger models tend to be harder to understand and lead to more errors than smaller models.

G2: Minimize the routing paths per element. When elements have a high degree in the model, i.e. many input and output routes, they make the model as a whole more difficult to understand, and designing models with high degree leads to errors in modeling.

G3: Use one start and one end event per pool. Higher numbers of start and end events increase the probability of errors in the model. Moreover, models that satisfy this guideline are easier to understand and allow for more forms of analysis to be made on them.

G4: Model as structured as possible, i.e., for each split there must be a join. A structured model matches respective split and join connectors of the same type. Unstructured models are more likely to include errors and harder to understand.

G5: Avoid OR routing elements. Ambiguities in the semantics of the OR-join lead to paradoxes and problems in implementation. Moreover, models that only use AND and XOR connectors are less prone to errors.

G6: Use verb-object activity labels. Verb-object labels (e.g. “Inhale infected droplets”) are considered less ambiguous and more useful than, for example, action-noun labels (e.g. ”Infected droplet inhalation”) or labels that follow neither style.

G7: Decompose the model if it has more than 50 elements. Related to G1, i.e., larger model sizes lead to more errors, it has been observed that for models with a number of elements higher than 50, error probability tends to be higher than 50% (MENDLING; REIJERS; AALST, 2010). For that reason, large models should be decomposed into collections of smaller models.

2.4 SARS-CoV-2 and COVID-19

According to Maier, Bickerton and Britton (2015) and Chen, Liu and Guo (2020), coronaviruses are the largest group of viruses of the *Nidovirales* order, which also includes *Arteriviridae*, *Coronaviridae*, *Mesonviridae* and *Roniviridae* families. The *Coronavirinae* are subdivided into alpha-, beta-, gamma-, and delta coronaviruses, by phylogenetic clustering (MAIER; BICKERTON; BRITTON, 2015). Coronaviruses are known to cause diseases of the respiratory, hepatic, nervous system, and gastrointestinal systems in humans. SARS-CoV-2, responsible for the COVID-19 pandemic, is a type of betacoronavirus. At first designated 2019-nCoV, The International Committee on Taxonomy of Viruses (GORBALENYA; BAKER; BARIC, 2020) later designated it SARS-CoV-2. It shares 96.2% similarity with the RaTG13 coronavirus, and 75.9% similarity to SARS-CoV, the coronavirus responsible for the 2002-2004 SARS outbreak (LUDWIG; ZARBOCK, 2020; WORLD HEALTH ORGANIZATION, 2003).

COVID-19 manifests with a wide clinical spectrum that ranges from asymptomatic patients to septic shock and multiorgan failure (CASCELLA et al., 2020). The WHO-China Joint Mission divided the clinical manifestations of the disease as *mild*, *moderate*, *severe*, and *critical* in their final mission report (WORLD HEALTH ORGANIZATION, 2020c). Other works have designated these severities as *light*, *common*, *moderate*, and *critical* (HUANG et al., 2020); however, we have elected to use WHO’s designations in our work, as advised by our domain expert.

Mild to moderate forms of the disease seem to represent the majority of cases, with moderate cases being characterized by the manifestation of a mild form of pneumonia. The WHO-China Joint Mission Report informs that symptom onset occurs on an average of 5 to 6 days after infection, with a range of 1 to 14 days, and that the time to clinical recovery for mild and moderate cases is approximately 2 weeks, and 3 to 6 weeks for patients with critical or severe disease. Later articles reported clearer divisions between severities, such as mild cases presenting with mild symptoms but no radiographic abnormalities, while moderate cases present with similar, albeit moderate symptoms and radiographic changes (WANG et al., 2020b; SINGH et al., 2020). Criteria for severe cases include dyspnea, respiratory frequency, blood oxygen saturation, and/or lung infiltrates of the lung field within 24-48h, and for critical cases include features of acute respiratory distress syndrome - ARDS, requiring mechanical ventilation, along with presence of multi-organ failure, metabolic acidosis and coagulation dysfunction.

The WHO-China Joint Mission did not report on symptoms by severity of disease; instead, it enumerated general signs and symptoms of infection. More recent publications, however, offer a more detailed view on symptoms by severity (HASSAN et al., 2020; SINGH et al., 2020). Mild cases may present with symptoms of an upper respiratory tract viral infection, which include dry cough, mild fever, nasal congestion, sore throat, headache, muscle pain, and malaise, without signs of a more serious disease, such as dyspnea, and can quickly deteriorate into severe or critical cases. Moderate cases present with respiratory symptoms of cough, shortness of breath, and tachypnea, but no signs of severe pneumonia. Severe cases present with severe dyspnea, acute respiratory distress syndrome - ARDS, sepsis, or septic shock (HASSAN et al., 2020). A small percentage of cases evolve, after about a week of infection, to critical disease with sudden worsening of clinical conditions, with respiratory failure, RNAemia (detectable presence of viral load in the blood), cardiac injury, septic shock, or multiple organ failure (CASCELLA et al., 2020; SINGH et al., 2020).

According to the WHO, from an identification and classification standpoint, cases are either *suspected*, *probable*, or *confirmed* (WORLD HEALTH ORGANIZATION, 2020b; KUMAR et al., 2020). A *suspected case* of COVID-19 infection can be characterized by three different options. The first option is a case in which the patient presents with severe acute respiratory illness - SARI (WORLD HEALTH ORGANIZATION, 2020b), which is an acute respiratory infection with history or active fever equal or greater than 38 ºC, with its onset within the last 10 days before being analyzed, and that requires hospitalization.

The second option is a case in which a person meets both the clinical and epidemiological criteria for suspected cases. The clinical criteria for a suspected case are:

- Acute onset of fever and cough;
- Acute onset of *any three or more* of the following signs or symptoms: fever, cough, general weakness or fatigue, headache, myalgia, sore throat, coryza, dyspnoea, anorexia, nausea, vomiting, diarrhoea, altered mental status.

The epidemiological criteria for a suspected case are:

- Resided or worked in a setting with a high risk of transmission for the SARS-CoV-2 virus, such as closed residential settings and humanitarian settings like camp and camp-like settings for displaced persons, at any time within the 14 days before symptom onset;
- Resided or traveled to an area with community transmission at any time within the 14 days before symptom onset;
- Worked in a health setting, including health facilities and households, anytime within the 14 days before symptom onset.

The third option is a case in which an asymptomatic person that does not meet the epidemiological criteria outlined above has tested positive for SARS-CoV-2 in an antigen-detecting rapid diagnostic test - Ag-RDT (WORLD HEALTH ORGANIZATION, 2020b).

A *probable case* of COVID-19 disease can be characterized by four distinct options. The first option is a case in which the patient meets the clinical criteria for suspected cases and is also a contact of a probable or confirmed case of COVID-19 disease, or is linked to a COVID-19 case cluster. The second option is a suspected case (as described above) for which chest imaging shows findings that suggest COVID-19 disease. The third option is a case in which a person presents with recent onset of anosmia or ageusia (loss of smell or taste) with no other identified cause. The fourth option is a death that could not be otherwise explained of an adult with respiratory distress preceding their death, and who was a contact of a probable or confirmed case of COVID-19 disease, or linked to a COVID-19 case cluster.

A *confirmed case* of COVID-19 disease can be characterized by three different options. The first option is that in which a person has tested positive for SARS-CoV-2 in a Nucleic Acid Amplification Test - NAAT (WORLD HEALTH ORGANIZATION, 2020b). The second option is a probable case (as described above) or a person matching the first two options for a suspected case who also tested positive for COVID-19 disease in an Ag-RDT. The third option is a case in which the person is asymptomatic, has tested positive for COVID-19 disease in an Ag-RDT, and is a contact of a probable or confirmed case of COVID-19 disease.

A *contact*, as defined by the WHO, is a person who had face-to-face contact with a probable or confirmed case within 1 meter or proximity, and for at least 15 minutes; or had direct physical contact with a probable or confirmed case; or directly cared for a patient with probable or confirmed COVID-19 disease without the use of the recommended protective equipment; or any other number of situations as indicated by local risk assessments, as described in (WORLD HEALTH ORGANIZATION, 2020b).

2.5 Related Works

We have divided the related works into two categories: BPM and BPMN applied to healthcare processes, and translation of BPMN 2.0 process models from and to natural language. They are presented in Table 2.1.

Table 2.1: Related Works divided by category

Categories	
BPM applied to Healthcare Processes	**BPMN models and Natural Language Text Descriptions**
Usefulness of BPM in Healthcare Settings • Fernández, Fernández and García (2020) • Reichert (2011) • Luciano, Pinto and Nunes (2020) (self-published)	*Extracting BPMN Models from Text* • Ferreira, Thom and Fantinato (2017) • Friedrich, Mendling and Puhlmann (2011)
Healthcare Process Modeling • Kopecky and Tomaskova (2020) • Ilahi, Ghannouchi and Martinho (2016)	*Extracting Natural Language Text from BPMN Process Models* • Leopold, Mendling and Polyvyanyy (2014) • Silva et al. (2019)

Source: the authors.

2.5.1 BPM and BPMN applied to Healthcare Processes

Several works report on the challenges and applications of BPMN 2.0 notation to healthcare processes, and their use to design process models related to specific diseases. Luciano, Pinto and Nunes (2020) present a study of the importance of process modeling in healthcare settings, with a brief review of the literature on the subject and a quantitative approach through a questionnaire applied to healthcare professionals in Portugal. Their results present a positive interest in the implementation of BPMN 2.0 notation in their field, even with most of these professionals not well-versed on the notation, due to the broader knowledge of the processes of their institutions and the promise of a quicker form of lookup than current procedure manuals.

Reichert (2011) discusses the possible impact of the application of BPM technologies to healthcare process support. The author points out issues resulting from the application domain of these processes, and discusses what functionalities are needed for Process-Aware Information Systems - PAISs (REICHERT; WEBER, 2012) to be useful to organizations and participants in these processes.

Fernández, Fernández and García (2020) present a systematic literature review on the application of the BPM discipline to clinical processes, analyzing their usefulness for improving the quality and efficiency of these processes. The results of this review point to qualitative improvements of the redesign or standardization of processes with BPMN notation and quantitative improvements related to process automation. It also reports benefits of applying BPM for the management and optimization of healthcare processes.

In Kopecky and Tomaskova (2020), the authors present the design of the treatment and care process for patients with Alzheimer's Disease in a hospital setting, with the goal of using these healthcare process models for cost simulations, quantitative prediction models, among other applications. Due to insufficient data, the authors were not able to design the progression of the disease (additionally to its treatment and care), as well as the process participants. Finally, the authors also do not describe the method used to design the process models.

Ilahi, Ghannouchi and Martinho (2016) present the design of a healthcare process for home healthcare in Tunisia. The information for their design was gathered via interviews and observation. The authors propose three sub-processes in their home healthcare process: patient admission, organizational care, and patient care, and point out patient care is a process that requires a more dynamic approach, as real cases vary amongst themselves.

These works present the application of BPM and BPMN 2.0 notation to the healthcare process domain, however all of them are applications to organizational processes, such as treatment evolution, shared decision of clinical approach, and supplemental care. In this work, we present the design of healthcare domain processes, but we do so in the context and mostly from the point of view of people infected by SARS-CoV-2 and manifesting symptoms of COVID-19. Moreover, due to the amount of knowledge generated towards fighting the pandemic, it was possible to model how the virus is transmitted and how the disease progresses in our case study.

2.5.2 Extraction of BPMN 2.0 Process Models to and from Natural Language Texts

Friedrich, Mendling and Puhlmann (2011) present an automated approach for generating models in BPMN from natural language texts. Their approach makes use of computational linguistics and Natural Language Processing - NLP tools and techniques to aid in the discovery of processes from these texts, and applies different levels of analysis (sentence, text) to gather information in a World Model. The information in the World Model is then used to generate a process model. The authors validated this approach with 47 pairs of texts and models from industry and textbook sources, achieving a 76.98% similarity between the given process models and those originated from their approach.

Ferreira, Thom and Fantinato (2017) present a semi-automatic approach to identifying process elements in natural language texts, defining a set of 32 rules to map and identify process elements in texts, separated by BPMN element type, based on different patterns of combined syntactic elements. Their approach was implemented in the form of a prototype tool, and in evaluating a set of 56 documents this approach achieved 91.92% accuracy. Their approach was also validated through a survey that showed 93.33% agreed with the proposed mapping rules.

Leopold, Mendling and Polyvyanyy (2014) propose an automatic technique for generating natural language texts from business process models, with a text generation approach that builds on natural language generation systems, using information that exists in the process models to generate text. The aim of their work was the support of process model validation by generating natural language representations of those models. The authors validate their approach via text-model pair evaluation and user evaluation and comparison. In this approach, the authors present Sentence Templates for transforming "Bonds", such as gateway splits and joins, that apply fixed forms to these points in the process.

Silva et al. (2019) present a service-oriented architecture that takes a process description written in natural language and creates a sound process description from it. Their approach also outputs a list of verification messages related to possible soundness issues in the model as a byproduct of the methodology. The authors implemented a prototype of their proposed architecture for validation, which was able to cover 95% of the information extracted from their original descriptions in average while maintaining soundness quality properties. This approach expands on the Sentence Template seen in Leopold, Mendling and Polyvyanyy (2014)'s approach to include other elements of the BPMN 2.0 notation.

In this work, we make use of the idea and sequence of analysis presented by

Friedrich, Mendling and Puhlmann (2011), combined with the rules proposed by Ferreira, Thom and Fantinato (2017), to manually create process models from natural language text, as described in section 3.1. From these models, by using an adaptation of sentence templates to the specific process domain, in our case healthcare, presented by Leopold, Mendling and Polyvyanyy (2014) and expanded on by Silva et al. (2019), we manually generate structured text descriptions for these process models. Afterwards, we validate these models and text descriptions with a domain expert, and make use of Petri Nets (REISIG, 1985) to ensure our process models are structurally correct, using the mapping technique proposed by Dijkman, Dumas and Ouyang (2008) to transform these BPMN models into structurally equivalent Petri Nets, and use the Woflan software (VERBEEK; AALST, 2000) for the analysis of different properties of these models, as presented in Chapter 4.

2.6 Chapter Summary

In this chapter, we presented the discipline of Business Process Management - BPM, giving emphasis to the Process Discovery phase of the BPM lifecycle. We also presented the Business Process Model and Notation - BPMN 2.0 notation in the context of the case study included in this work, describing what each of the elements represent in the process model, and the concepts of Process Model Quality. Moreover, we presented an overview of the subjects of this case study, the SARS-CoV-2 virus and the COVID-19 disease, as found in the literature, and finally presented related works divided in two categories: BPM and BPMN applied to healthcare processes, and the extractions of process descriptions from BPMN process models and of process models from natural language process descriptions.

3 AN APPROACH FOR PROCESS MODEL EXTRACTION FROM SPREAD OUT NATURAL LANGUAGE DOCUMENTS

This chapter presents the scientific methodology of the process-oriented approach we developed for extracting process models from spread out documentation. The approach presented here is based on the use of systematic mapping for information gathering (PETERSEN; VAKKALANKA; KUZNIARZ, 2015), designing BPMN process models from natural language documents (FRIEDRICH; MENDLING; PUHLMANN, 2011; FERREIRA; THOM; FANTINATO, 2017), and extracting text descriptions from BPMN process models (LEOPOLD; MENDLING; POLYVYANYY, 2014; SILVA et al., 2019). Then, we evaluate the results of this approach with a case study on SARs-CoV-2 and COVID-19 processes.

3.1 Process-Oriented Approach

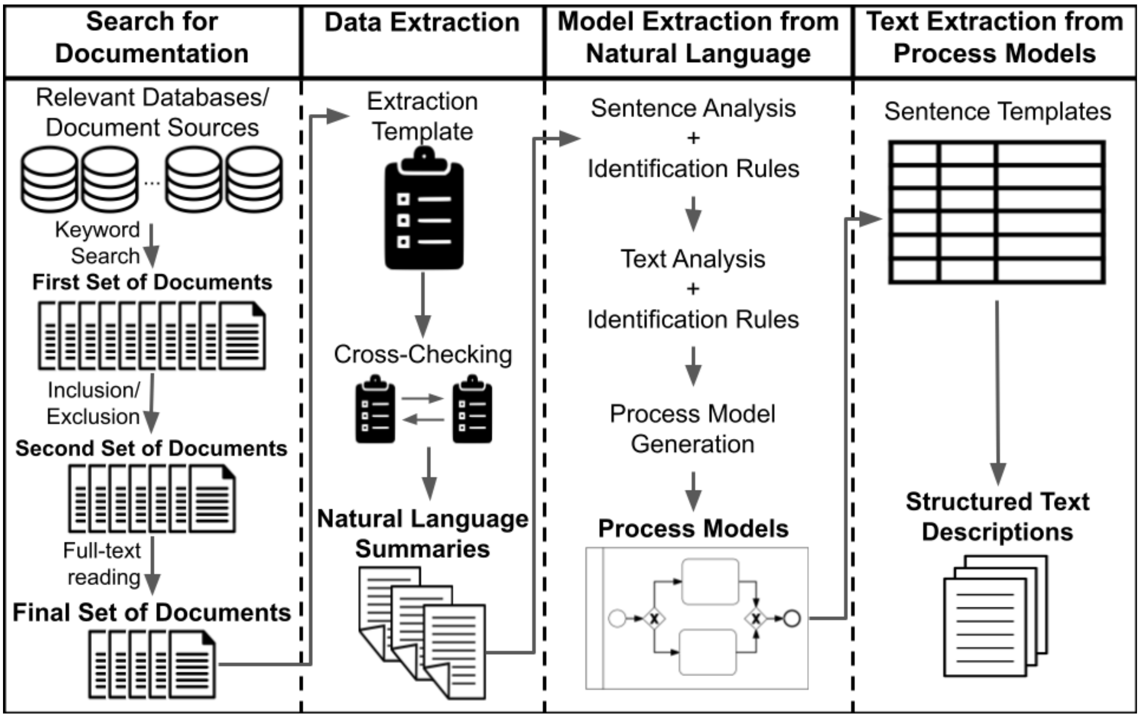
We divide the approach we developed to extracting process models from spread out documentation into a methodology including the *search for documentation*, *data extraction*, *model extraction from natural language*, and *text extraction from process models*. Figure 3.1 illustrates the approach.

Search for Information, represented by the first column of Figure 3.1, is concerned with finding documentation related to the processes to be designed, but also with ensuring that the documents found are useful for process design purposes. We employ systematic mapping principles and strategies (PETERSEN; VAKKALANKA; KUZNIARZ, 2015) to achieve the goal of this phase.

With knowledge of the process domain, we must define what are reputable sources of documentation for that domain. These sources should include any databases and sources of documents for the process domain, such as scientific article indexers and authority institutions' document repositories, that are good candidates for trustworthy and information-rich documentation.

With the sources of documentation defined, we must conduct an initial search in these sources, making use of a keyword set that represent more closely the processes that are to be designed. This initial search will produce a first set of documents. This first set of documents must, then, be trimmed according to inclusion and exclusion criteria. These criteria are crucial for discarding outdated information, and ensuring useful and correct information are kept in the second set of documents that is the result of the application of the established criteria.

Figure 3.1: Methodology developed for discovery of Processes from spread out documentation



Source: The authors

After concluding this first round of exclusions from the document set, we must then read the full texts of the remaining documents to ensure they are related to the processes to be designed and have information that can be used for that. This step is the beginning of the application of document analysis for process discovery. After full-text reading, the documents are determined to either be useful for process design, or not. Those that are deemed useful form the final document set, from which data on the processes to be designed are extracted.

Data Extraction has the goal of extracting the information from the set of documents obtained in the previous phase and generating a summary of that information to inform process model design in the following phase of the method. Data extraction is also guided by principles and tools described in Petersen, Vakkalanka and Kuzniarz (2015).

We develop and make use of an extraction template for gathering information from the documentation assembled. An extraction template, in the context of this work, is a collection of questions to drive the extraction of useful information from the document about the subject of our search. This extraction template must be developed in a process-oriented fashion, so it facilitates the creation of a process-oriented summary later on.

Identifying actors, starting conditions, and end conditions, as well as any other important process markers, such as decision or exclusive choice points, in the template make it so that compiling a good summary becomes easier and provides a better starting point for process design afterwards. An example of such a template is presented in our case study, in Section 3.2.

After applying the extraction template to each document in the set, we cross-check similar data between them, both in order to ensure correctness and to resolve conflicts between different documents. If there are documents in the final set that are from authority agencies or organizations, the extracted data from these documents are considered to be "correct", and data conflicting with them are discarded before the information summary is assembled. If domain experts are available for consultation they should be employed as an added resource, but they are not an integral part of this method.

Having cross-checked all data extracted from the documentation that was gathered, one or more summaries are then composed from all the information available. These summaries are written in natural language, but they should be composed in a way that facilitates process design. When writing these summaries, laying out the steps of the process, as extracted from documentation, in sequential order may allow for a more correct model in the following phase of the method; clearly stating the actors of those steps is also good practice as one composes the summaries. The summaries assembled at the end of data extraction are used as entries for process model generation.

Model Extraction from Natural Language produces process models in BPMN 2.0 notation from the natural language summaries created in the previous phase. This phase is inspired by Friedrich, Mendling and Puhlmann (2011), and supported by Ferreira, Thom and Fantinato (2017).

At this point in the approach, we apply sentence level analysis, and then text level analysis (FRIEDRICH; MENDLING; PUHLMANN, 2011) to the natural language summaries. In sentence analysis, we determine what are the actors and activities (actions) being described, and using the patterns from Ferreira, Thom and Fantinato (2017) identify what BPMN 2.0 elements are being described in natural text. These patterns are presented in Appendix A. In text analysis, we determine the relationship sentences have with each other, in order to connect tasks and decision points in the correct order and in correct relation between themselves (FRIEDRICH; MENDLING; PUHLMANN, 2011). The rules described in Ferreira, Thom and Fantinato (2017) are used here as well.

Having the elements identified and their relationships with one another laid out, we generate one of more process models, depending on how many summaries and processes were identified and discovered from the information gathered, which then are used to generate structured process descriptions.

Text Extraction from Process Models takes the process models designed in BPMN 2.0 and extracts from them structured descriptions in natural language. This phase is adapted from Leopold, Mendling and Polyvyanyy (2014) and Silva et al. (2019).

To generate structured text descriptions from process models, we use a sentence template table, such as the one introduced by Leopold, Mendling and Polyvyanyy (2014) and later expanded upon by Silva et al. (2019), making adaptations so that it is useful for the context of the processes, in order to produce sound process descriptions. This sentence template table relates elements of the BPMN 2.0 notation and their uses, or bonds, in the process model to specific sentence templates to be applied for the extraction of the process description. There may be multiple options of sentence templates for the same form of use. We present a modified sentence template table in our case study, in Section 3.2.

We apply the templates to the elements and bonds in the process model in order of appearance in the model, that is, by beginning with the starting element in the model and following all paths in the process model until all elements have been extracted to text form.

3.2 Case Study

This section presents a case study demonstrating the application of the approach outlined in section 3.1 to the discovery of SARS-CoV-2 and COVID-19 processes. It reports on the progress of the case study as the approach was executed, and finally presents the process models and text descriptions for SARS-CoV-2 Transmission and Contagion processes and COVID-19 Symptomatic Manifestation and Identification processes.

3.2.1 Search for Documentation

Knowing that the processes we wanted to model were in the healthcare process domain, we identified the World Health Organization as the highest authority agency on healthcare. Therefore, all documents pertaining to SARS-CoV-2 and COVID-19 from the WHO were added to the initial document set. We also included Brazil's Health Ministry webpage (MINISTÉRIO DA SAÚDE, 2021) in the documents, as each country established their own protocols to be paired with WHO's guidelines for identification of COVID-19 infection, and we decided to focus on Identification in our country. Furthermore, having the WHO as a guideline, we identified other indexers and databases, such as Elsevier, PubMed, JAMA Network, Wiley, Springer Nature, as well as WHO's own indexer for SARS-CoV-2 and COVID-19 research.

Having that, we defined our main research question on this part of the approach as "What is known about SARS-CoV-2 and COVID-19?", which was then further broken down, to better drive data extraction later in the approach, into the following research questions (RQs):

- RQ1: What is known about the transmission and contagion of SARS-CoV-2?
- RQ2: What is known about the symptomatic progression of COVID-19?
- RQ3: What is known about the identification of a COVID-19 infection?

To search for this information, we used the search phrase

"("2019-nCoV" OR "COVID-19" OR "SARS-CoV-2") AND
("transmission" OR "spread" OR "clinical features" OR "features" OR "review")"

to execute the search in those databases. To ensure we had relevant documents in our final set, we excluded from our search results all documents that were not in English, that had not been indexed and peer reviewed, and that were published before 2020 (given the outbreak of SARS-CoV-2 happened at the end of 2019). After removing duplicates we still had over 100 documents in our set, so it was necessary to filter it further via abstract reading before entering the full-text reading step, which reduced it to 43 documents. This initial search was performed in July 2020, and from October 2020 onwards it was performed every 15 days to ensure the information we were using was as up-to-date as possible.

In the full-text reading, we searched for information that would help drive process design towards Transmission and Contagion of the SARs-CoV-2 virus, the Disease Progression and Manifestation of the COVID-19 infection, and the Identification of a COVID-19 infection. At the end of full-text reading, we had identified 12 documents with information that could aid in designing these processes.

3.2.2 Data Extraction

For extracting data from these documents, we developed the following extraction template (PETERSEN; VAKKALANKA; KUZNIARZ, 2015), shown in Table 3.1. This extraction template was created with the goal of extracting information for process design, focusing on the activities the documents describe, their actors, and the order they are described in, helping establish a preliminary order for tasks even before natural language summaries are assembled, so that it becomes easier to form a process-oriented summary afterwards. The data extraction was not exclusive to the documents' texts, but also encompassed tables and images present in them.

Table 3.1: Extraction Template for the Case Study.

Data Item	Value
General	
Document ID	Integer
Document Title	Name of the Document
Author Name	Set of Names of the Authors
Year of Publication	Calendar year
Process Information	
Information on RQs	Set of RQs the document has information on
Activities Described	Set of activities the document describes for each RQ
Order of Activities	List with order of described activities
Actors in Activities	List of actors for each described activity
Decision Points	List of major decision points described by the document
Split Points	List of split (parallel) points described by the document

Source: the authors

After the extraction template was applied for all documents, the extracted data was then cross-checked. Extracted data from the WHO was set as the standard for cross-checking, meaning that any information that was in disagreement with the data from WHO documents was ignored or discarded. Once cross-checking of the extracted data was finished, the natural language summaries were assembled. Three summaries were written, one for each research question. These summaries were compiled manually.

SARS-CoV-2 Transmission and Contagion

"For the spread of the SARS-CoV-2 virus to happen, there must be a person infected with the virus involved. The virus can be spread directly, indirectly, through the air, or through some other, yet unknown form of spread.

For the virus to spread directly, an infected person may touch another person while having their hands covered with virus particles, which deposits these particles on the healthy person, who then touches them and, afterwards, touches their own mouth, nose or eyes, becoming infected; or an infected person may also cough, sneeze or exhale close to another person, thus launching infected droplets through the air at a healthy person, who then touches these droplets and, afterwards, touches their own mouth, nose or eyes, becoming infected, or inhale these particles as they are suspended in the air, becoming infected.

For the virus to be spread indirectly, an infected person may touch an object or surface with their hands covered in virus particles, or cough over an object or surface, or sneeze over an object or surface, or exhale over an object or surface, all of which deposit droplets with virus particles on such an object or surface. Afterwards, a healthy person that touches this object or surface, given the virus is still active on it, may touch their mouth, nose or eyes, becoming infected.

For the virus to be spread airborne, an infected person must be subjected to an Aerosol Generating Procedure - AGP, which launches virus particles in the air in aerosols. A healthy person may then inhale those aerosolized particles, becoming infected.

It is not known how else the SARS-CoV-2 virus may spread, though there are theories being tested; if there are more forms of spread, an infected person and a healthy person must partake in such a form in order for the virus to be spread."

COVID-19 Symptomatic Progression

"The symptomatic progression of a COVID-19 infection starts with a person getting infected. The virus goes through an incubation period that may vary from 1 to 14 days, but has a mean of 5-6 days, before symptoms actually manifest. When they do, these symptoms may start out mild, moderate, or severe.

Mild symptoms are dry cough, mild fever, nasal congestion, sore throat, headaches, muscle pain, among others. The usual recovery period for someone with mild symptoms is of 2 weeks from symptom onset, that is, when symptoms started. The infected person may also die while manifesting these symptoms. Moderate symptoms are high fever, productive cough, shortness of breath, tachypnea (rapid breathing), mild pneumonia, and other symptoms. The usual recovery period for someone with moderate symptoms is of 2 weeks from the start of the symptoms. The infected person may also die while presenting these symptoms. Severe symptoms are severe pneumonia, Acute Respiratory Distress

Syndrome - ARDS, sepsis and septic shock, and other symptoms. The usual recovery period for someone with severe symptoms is of 3 to 6 weeks from symptom onset. The infected person may also die while presenting these symptoms.

While presenting with mild, moderate, or severe symptoms, an infected person's condition may deteriorate. From mild symptoms, it may deteriorate to severe or critical symptoms; from moderate or severe symptoms, it may deteriorate only to critical symptoms. Critical symptoms of a COVID-19 infection are respiratory failure, RNAemia (viral content flowing in the bloodstream), cardiac injuries, septic shock, Multiple Organ Dysfunction, and other symptoms. The usual recovery period for a person with critical symptoms is of 3 to 6 weeks. The infected person may also die while presenting these symptoms."

COVID-19 Identification

"For a person to identify if they have a COVID-19 infection or not, first they must check if they are presenting symptoms compatible with the disease, such as cough, fever, sore throat and/or coryza. If they are, they must avoid contact with other people and immediately seek a triage center.

A medical professional evaluating a case for COVID-19 infection must determine, first of all, if a person is a suspect case of COVID-19 or not. Several factors allow them to check that. A person is a suspect case of the infection if they have recently presented with coughing and fever, or suddenly contracted any three or more of the following symptoms: fever, cough, general weakness/tiredness, headaches, myalgia (muscle pain), sore throat, coryza, dyspnea (shortness of breath), anorexia/nausea/vomiting, diarrhoea, or altered mental status; AND resided or worked in a location with a high risk of virus transmission in the previous 14 days, OR resided or worked in an area with community transmission in the previous 14 days, or worked in a health setting in the previous 14 days. Another possibility for a suspect case is a patient presenting with SARI (Severe Acute Respiratory Illness, an acute respiratory infection with history of, or active, fever equal to or greater than 38 ºC onset within the 10 previous days before being checked). Another possibility for a suspect case is a person presenting no symptoms (asymptomatic) but for whom a COVID-19 quick-test tested positive.

If the person (now patient) is a suspect case, they might be considered a probable case if chest imaging also shows signs of COVID-19 infection. Another possibility for a person to be considered a probable case is one that presents with the symptoms described for a suspect case AND had contact with a probable or confirmed case of COVID-19 infection. Another possibility for a probable case is a person who has had a sudden loss of palate and sense of smell (ageusia and anosmia) that has no other possible cause.

A suspect case may be considered a confirmed case if they present the clinical symptoms and test positive for a COVID-19 quick test. another possibility for a confirmed case is an asymptomatic person, who tested positive on a COVID-19 quick-test, that is a contact of a probable or confirmed case. Finally, any person with a positive NAAT test is a confirmed case."

3.2.3 Model Extraction from Natural Language

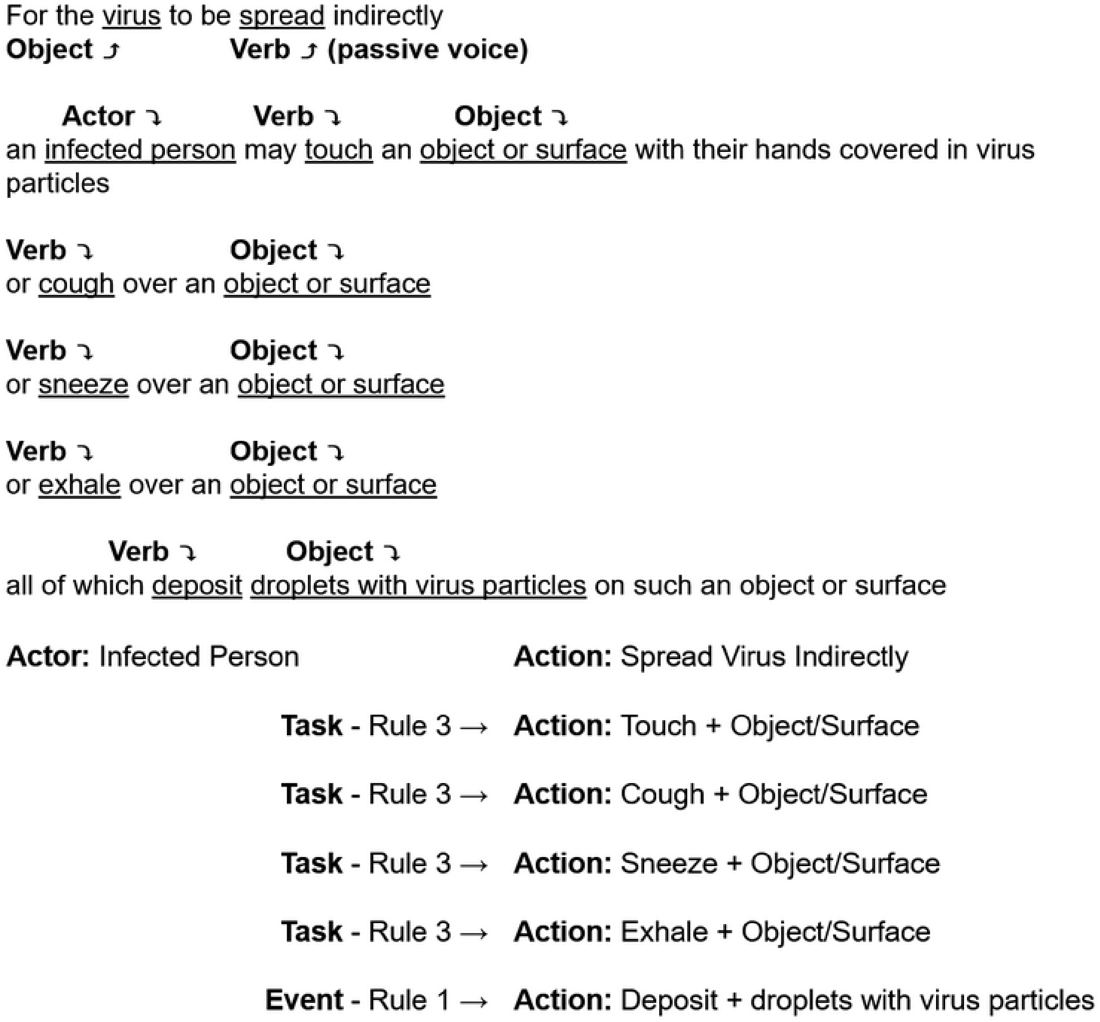
Having the natural language summaries, we manually applied sentence level analysis to them, identifying actors and actions being described in them. We applied the following steps for sentence analysis: 1) Split text into individual sentences; 2) Split sentences into phrases; 3) Extract Actors and Verbs; 4) Extract Object and Combine with Verb; 5) Combine Actors with Actions.

This task is also aided by the format developed for the extraction template in subsection 3.2.2, which already registers the actors observed in data extraction from the documents. An example of the application of these steps on the sentence *"For the virus to be spread indirectly, an infected person may touch an object or surface with their hands covered in virus particles, or cough over an object or surface, or sneeze over an object or surface, or exhale over an object or surface, all of which deposit droplets with virus particles on such an object or surface"*, from the *SARs-CoV-2 Transmission and Contagion* summary, can be seen in Figure 3.2.

After Sentence Level Analysis, Text Level Analysis was manually applied to the summaries, to identify the relations between sentences. The first step in this analysis is to detect conditional markers, words or expressions that show characteristics that can be mapped to different BPMN constructs. Examples of conditional markers are underlined in the following paragraph of the *COVID-19 Identification* summary:

"If the person (now patient) is a suspect case, they might be considered a probable case if chest imaging also shows signs of COVID-19 infection. Another possibility for a person to be considered a probable case is one that presents with the symptoms described for a suspect case AND had contact with a probable or confirmed case of COVID-19 infection. Another possibility for a probable case is a person who has had a sudden loss of palate and sense of smell (ageusia and anosmia) that has no other possible cause."

Figure 3.2: Sentence Level Analysis Example



For the virus to be spread indirectly
Object ⤴ **Verb ⤴ (passive voice)**

Actor ⤵ **Verb ⤵** **Object ⤵**
an infected person may touch an object or surface with their hands covered in virus particles

Verb ⤵ **Object ⤵**
or cough over an object or surface

Verb ⤵ **Object ⤵**
or sneeze over an object or surface

Verb ⤵ **Object ⤵**
or exhale over an object or surface

Verb ⤵ **Object ⤵**
all of which deposit droplets with virus particles on such an object or surface

Actor: Infected Person **Action:** Spread Virus Indirectly

Task - Rule 3 → **Action:** Touch + Object/Surface

Task - Rule 3 → **Action:** Cough + Object/Surface

Task - Rule 3 → **Action:** Sneeze + Object/Surface

Task - Rule 3 → **Action:** Exhale + Object/Surface

Event - Rule 1 → **Action:** Deposit + droplets with virus particles

Source: The authors

After having identified the condition markers, we combined information contained in different Actions, and generated flows between Actions to form the process models. The process models resulting from the application of these steps to the case study can be found in subsection 3.3.

3.2.4 Text Extraction from Process Models

Having the process models designed in BPMN 2.0 notation, we can extract text descriptions from them, in natural language, that use language that seeks to be unambiguous. To do that, we adapted the approach presented by Leopold, Mendling and Polyvyanyy (2014) and expanded upon by Silva et al. (2019) of applying sentence templates to extract text descriptions from process models. We adapted the sentence template table from Silva et al. (2019) for our work, as the sentence templates were not comprehensive enough for our process domain. Our adaptation is presented in table 3.2. We manually applied these templates to the process models of our case study in order to extract the descriptions that accompany them in section 3.3.

Table 3.2: Sentence Templates applied in this work to compose sound process descriptions.

Element	Type	Sentence Templates
Start	Sequence	The <*process*> process starts when...
Sequence	Atomic 1 Atomic 2 Atomic 3	Then, ... After <that | *illness* manifests>, ... Subsequently, ...
End	Sequence	The process ends with, ...
Exclusive Choice	Split 1 Split 2 Join	The <*cond.*>may either be <*first*>, or <*second*>, ... The <*role*>may either be <*first*>, or <not | *second*>, ... In any <case | of these cases>, ...
Inclusive Choice	Split Join	... <*number*>alternative procedures are executed in an arbitrary order. Afterwards, ...
Choice Paths	Ordinal Conditional	- If <*condition*>, ...
Parallelism	Split Join Path 1 Path 2	..., <*number*>procedures are executed in an arbitrary order. After each case, ... In the meantime, ... At the same time, ...
Loop	Join 1 Join 2	If required, <*role*>repeats the latter steps and continues with ... Once the loop is finished, ...

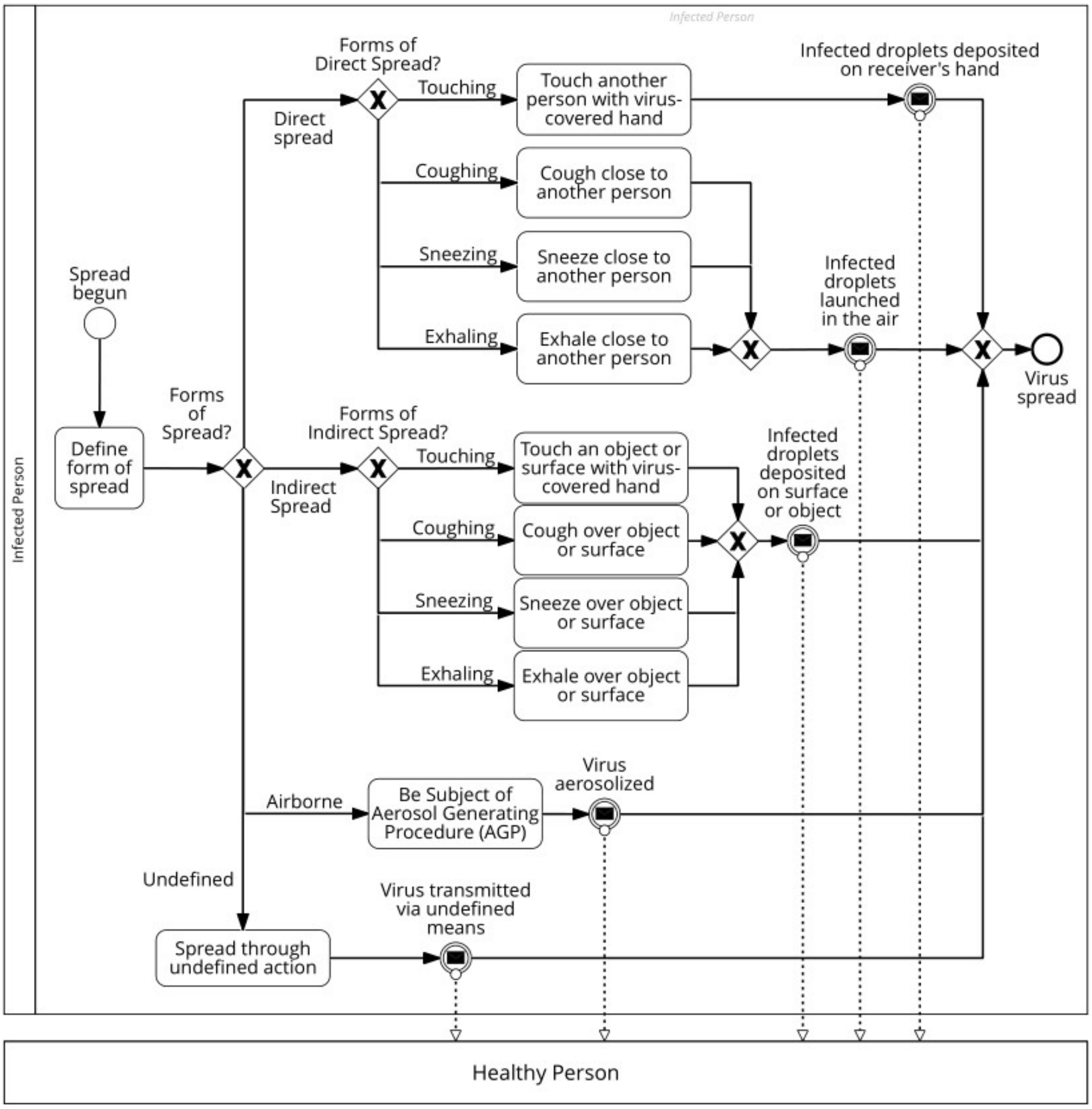
Source: Adapted from Silva et al. (2019)

3.3 Resulting Process Models and Text Descriptions

In this subsection, we have gathered the resulting process models and text descriptions for SARS-CoV-2 Transmission, SARS-CoV-2 Contagion, COVID-19 Symptomatic Manifestation, and COVID-19 Identification. The first two processes were extracted from the *SARS-CoV-2 Transmission and Contagion* summary; COVID-19 Symptomatic Manifestation was extracted from the *COVID-19 Symptomatic Progression* summary; and COVID-19 Identification was extracted from its eponymous summary.

3.3.1 SARS-CoV-2 Transmission

Figure 3.3: *SARS-CoV-2 Transmission* process model



Source: The authors

The *SARS-CoV-2 Transmission* (Figure 3.3) process can start at any point in the infectious period of the disease for an infected person (*Start Event: Spread begun*). The infected human defines in which form they will spread the virus (*Task: Define form of spread*). The virus can either be spread through direct spread (person to person), or through indirect spread (fomites), or through airborne spread, or through another undefined form of spread.

If the virus is spread through direct spread, then this direct spread can either be through touching, coughing, sneezing, or exhaling. If the virus is spread through touching, then the infected person touches another person with their virus-covered hands (*Task: Touch another person with virus-covered hands*), which leads to infected droplets being deposited on the receiver's hands (*Intermediate Message Throwing Event: Infected droplets deposited on receiver's hand*). If the virus is spread through coughing, then the infected human coughs close to another person (*Task: Cough close to another person*); if the virus is spread through sneezing, then the infected human sneezes close to another person (*Task: Sneeze close to another person*); if the virus is spread through exhaling, then the infected human exhales close to another person (*Task: Exhale close to another person*). If the virus is spread through coughing, sneezing, or exhaling, these then lead to infected droplets being launched in the air (*Intermediate Message Throwing Event: Infected droplets launched in the air*).

If the virus is spread through indirect spread, then this indirect spread can either be through touching, coughing, sneezing, or exhaling. If the virus is spread through touch, then the infected human touches an object or surface with their virus-covered hands (*Task: Touch an object or surface with virus-covered hands*); if the virus spread through coughing, then the infected human coughs over an object or surface (*Task: Cough over object or surface*); if the virus is spread through sneezing, then the infected human sneezes over object or surface (*Task: Sneeze over object or surface*); if the virus is spread by exhaling, then the infected human exhales over an object or surface (*Task: Exhale without covering mouth over object or surface*). In any case, it leads to infected droplets being deposited or a surface or an object (*Intermediate Message Throwing Event: Infected droplets deposited on surface or object*).

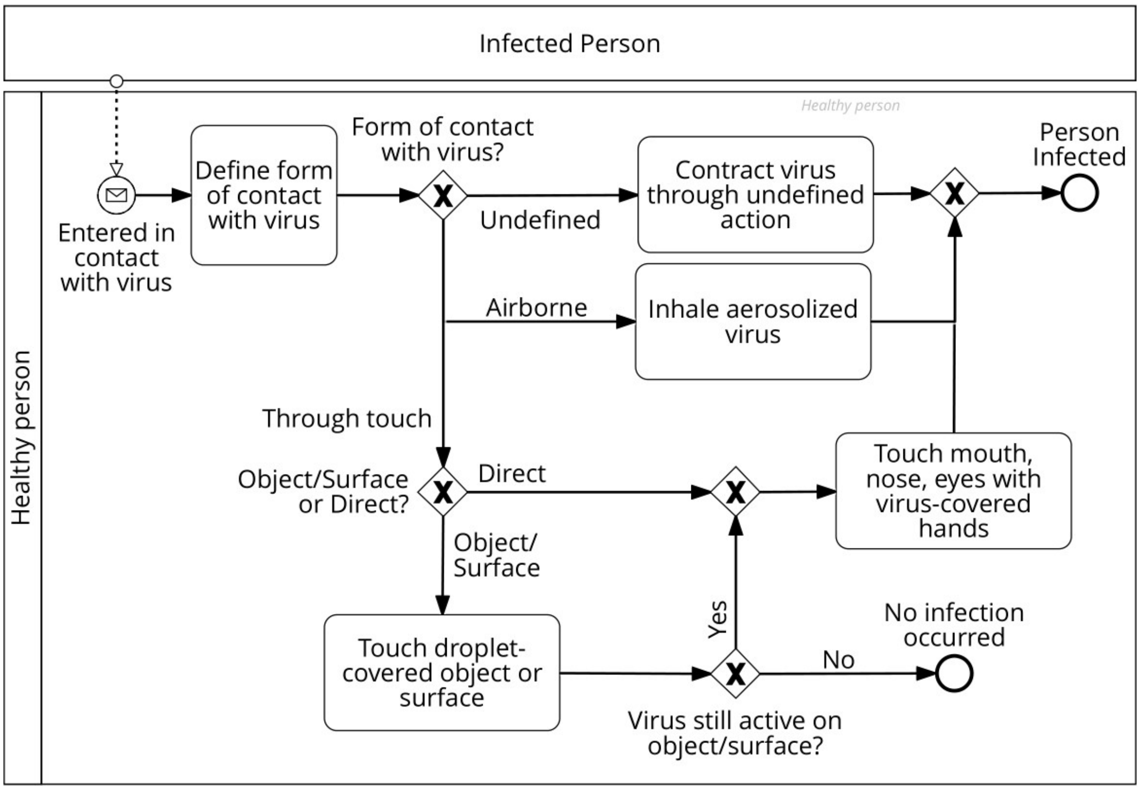
If the virus is spread through airborne transmission, then the infected person is subjected to an Aerosol Generating Procedure (AGP) (*Task: Be Subject of Aerosol Generating Procedure*).

If the virus is spread through an undefined form of spread, then the infected human spreads it with an undefined action (*Task: Spread through undefined action*). After that, the virus is spread through undefined means (*Intermediate Message Throwing Event: Virus transmitted via undefined means*).

In any of these cases, the process ends when the virus is spread (*End Event: Virus spread*).

3.3.2 SARS-CoV-2 Contagion

Figure 3.4: *SARS-CoV-2 Contagion* process model



Source: The authors

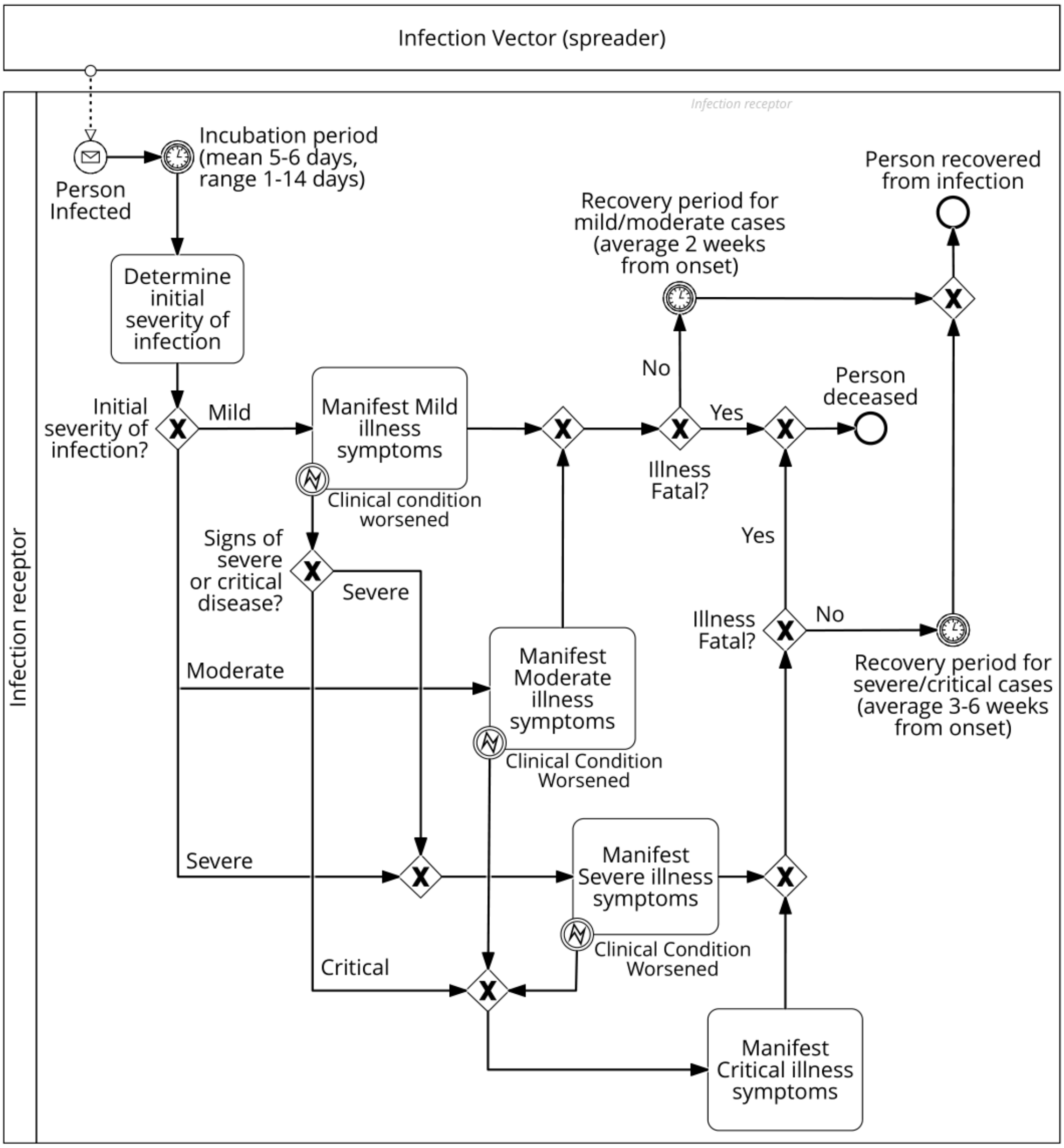
The *SARS-CoV-2 Contagion* process starts when a healthy person enters in contact with the virus (*Start Message Event: Entered in contact with virus*). Then, it is determined in what form they do so (*Task: Define form of contact with virus*). A healthy person can enter in contact with the virus either through an undefined form, or through airborne virus, or through touch.

If the form of contact with the virus is undefined, then the healthy human contracts the virus through an undefined action. (*Task: Contract virus through undefined action*). After that, the process ends with the person being infected (*End Event: Person infected*).

If the form of contact with the virus is through airborne virus, then the healthy human inhales aerosolized virus (*Task: Inhale Aerosolized Virus*). If the form of contact with the virus is through touch, then it can be either by directly touching the virus on another person (direct) or on an object or surface. If the virus is touched in a direct manner, then the healthy human touches their mouth, nose, or eyes with virus-covered hands (*Task: Touch mouth/nose/eyes with virus-covered hands*). If the virus is touched on an object or surface (indirectly), then the healthy human touches the infected-droplet-covered object or surface (*Task: Touch droplet-covered object or surface*). The virus can either still be active or expired on the object or surface. If the virus is still active, then the healthy human touches their mouth, nose or eyes with virus-covered hands (*Task: Touch mouth/nose/eyes with virus-covered hands*). In any of these cases, the process ends with the person being infected (*End Event: Person infected*). If the virus is expired, then the process ends with no infection (*End Event: No infection occurred*).

3.3.3 COVID-19 Symptomatic Manifestation

Figure 3.5: *Symptomatic Manifestation of COVID-19 Disease* process model



Source: The authors

The *Symptomatic Manifestation of COVID-19 Disease* process (Figure 3.5) starts when a person is infected (*Start Event: Person Infected*). Then the infection receptor, or infected receptor, goes through the disease incubation period before manifesting any symptoms (*Intermediate Timer Event: Incubation period*). This period may range from 1 to 14 days in duration, with a mean duration of 5 to 6 days. After that, the disease may initially manifest with different severities (*Task: Determine initial severity of infection*). The initial severity can either be mild, moderate, or severe.

If the severity of the disease is mild, then the infected receptor goes through the mild symptomatic manifestation of the illness (*Task: Manifest Mild illness symptoms*). In the mild symptomatic manifestation of the disease, the infected receptor may develop a range of mild symptoms: dry cough, mild fever, nasal congestion, sore throat, headache, muscle pain, malaise, or other mild symptoms. While manifesting mild symptoms, the infected receptor may experience a worsening of their clinical condition (*Intermediate Boundary Error Event: Clinical Condition Worsened*). If they do, their condition may deteriorate to either severe or critical disease. If it deteriorates to severe disease, the infected receptor goes through the severe symptomatic manifestation of the disease, which is explained later in the process description. If they deteriorate to critical disease, they go through the critical symptomatic manifestation of the disease, which is explained later in the process description.

If the severity of the disease is moderate, then the infected receptor goes through the moderate symptomatic manifestation of the disease (*Task: Manifest Moderate Illness symptoms*). In the moderate symptomatic manifestation of the disease, the infected receptor may develop a range of moderate symptoms: fever, cough, shortness of breath, tachypnea, or other moderate symptoms. While manifesting moderate symptoms, the infected receptor may experience a worsening of their clinical condition (*Intermediate Boundary Error Event: Clinical Condition Worsened*). If they do, their condition deteriorates to critical disease, and they go through the critical symptomatic manifestation of the disease, which is explained later in the process description.

After either mild or moderate illness manifests, it may either be fatal to the infected receptor, or not. If it is, the process ends when the person is deceased (*End Event: Person deceased*). If not, they go through the recovery period for mild or moderate cases (*Intermediate Timer Event: Recovery period for mild/moderate cases*), and then the process ends when they are recovered from the infection (*End Event: Person recovered from infection*).

If the disease symptoms are severe, then the infected receptor goes through the severe manifestation of the disease (*Task: Manifest Severe Illness Symptoms*). In the severe symptomatic manifestation of the disease, the infected receptor may develop a range of severe symptoms: severe pneumonia, ARDS, sepsis, septic shock, or other severe symptoms. While manifesting severe symptoms, the infected receptor may experience a worsening of their clinical condition (*Intermediate Boundary Error Event: Clinical Condition Worsened*). If they do, their condition deteriorates to critical disease, and they go through the critical symptomatic manifestation of the disease, which is explained later in the process description.

If the infected receptor deteriorates to critical disease, then they go through the critical symptomatic manifestation of the disease (*Task: Critical Illness Symptomatic Manifestation*). In the critical symptomatic manifestation of the disease, the infected receptor may develop a range of critical symptoms: respiratory failure, RNAemia, cardiac injury, septic shock, Multiple Organ Dysfunction, or other critical symptoms.

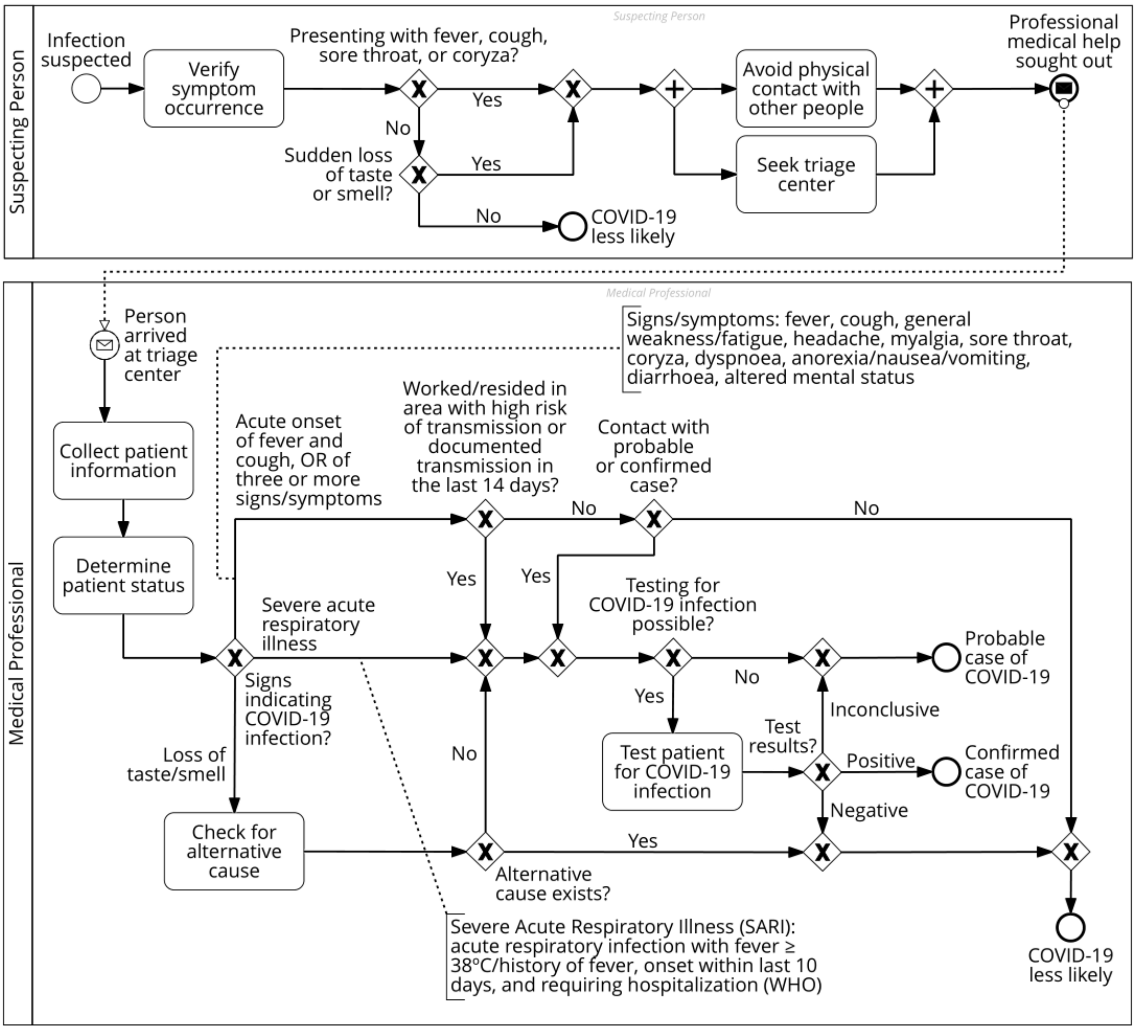
After either severe or critical illness manifests, it may either be fatal to the infected receptor, or not. If it is, the process ends when the person is deceased (*End Event: Person deceased*). If not, they go through the recovery period for severe or critical cases (*Intermediate Timer Event: Recovery period for severe/critical cases*), and then the process ends when they are recovered from the infection (*End Event: Person recovered from infection*).

3.3.4 COVID-19 Identification

The *Identification of COVID-19 Disease* process (Figure 3.6) starts when a person suspects infection (*Start Event: Infection Suspected*). Then, the person verifies symptom occurrence (*Task: Verify symptom occurrence*). The person may be presenting with fever, cough, sore throat, or coryza, or not. If they are not, they may be presenting with sudden loss of taste or smell, or not. If they are not, the process ends with COVID-19 being less likely (*End Event: COVID-19 less likely*).

If the person is presenting with the above symptoms or has had sudden loss of taste or smell, then they avoid physical contact with other people (*Task: Avoid physical contact with other people*). At the same time, they seek a triage center (*Task: Seek triage center*).

Figure 3.6: *Identification of COVID-19 Disease* process model



Source: The authors

After professional medical help is sought out (*Message End Event: Professional medical help sought out*), a medical professional receives the person at the triage center (*Message Start Event: Person arrived at triage center*), subsequently collecting their information (*Task: Collect patient information*), and then determines the person's status (*Task: Determine patient status*). The signs indicating COVID-19 infection may either be acute onset of fever and cough or three or more signs/symptoms of the disease, or severe acute respiratory illness, or loss of taste/smell. If the patient has acute onset of fever and cough or three or more signs/symptoms of the disease, they may either have worked or resided in an area with high risk or documented case of transmission in the last 14 days, or not. If the patient has not, they may either have had contact with a probable or confirmed case of COVID-19 infection, or not. If the patient has experienced loss of taste/smell, the medical professional checks for an alternative cause (*Task: Check for alternative cause*).

There may either be an alternative cause, or not. If the patient has worked or resided in an area as was described, or had contact with a probable or confirmed case, or has severe acute respiratory illness, or had loss of taste/smell with no alternative cause, they are a suspect case of COVID-19.

If the patient has not had contact with a probable or confirmed case, or there is an alternative cause for their loss of taste/smell, the process ends with COVID-19 being less likely (*End Event: COVID-19 less likely*).

If the patient is a suspect case of COVID-19, testing for COVID-19 infection may either be possible, or not. If it's not possible, the process ends with the patient being a probable case of COVID-19 (*End Event: Probable case of COVID-19*). If it is possible, the medical professional tests the patient for COVID-19 infection (*Task: Test patient for COVID-19 infection*). The test results may either be inconclusive, positive, or negative. If the test result is inconclusive, the process ends with the patient being a probable case of COVID-19 (*End Event: Probable case of COVID-19*). If it is positive, the process ends with the patient being a confirmed case of COVID-19 (*End Event: Confirmed case of COVID-19*). If the test is negative, the process ends with COVID-19 being less likely (*End Event: COVID-19 less likely*).

3.4 Chapter Summary

In this chapter, we presented a process-oriented approach developed to extract process models from spread out, natural language documentation, divided into *Search for Information*, *Data Extraction*, *Model Extraction from Natural Language*, and *Text Extraction from Process Models*. Subsequently, we presented a case study of this approach on SARS-CoV-2 and COVID-19 processes, describing the steps taken as well as a sentence template table adapted to the process domain of the case study, and finally presented the resulting process models and text descriptions of the case study.

4 VALIDATION AND VERIFICATION OF THE CASE STUDY RESULTS

In this chapter, we present the semantic validation of the process models generated in our case study, as well as their structural verification by BPMN experts. We also present Petri Nets (REISIG, 1985) and an approach to mapping BPMN process models to them, and the verification of these mapped nets with the use of Woflan, a Petri-net-based analysis tool(VERBEEK; AALST, 2000), to verify the syntactic quality of our process models through analysis of both static and behavioral properties. Validation and verification of process models are forms to measure their quality (section 2.3).

4.1 Semantic Validation of the Processes

To semantically validate the processes that were designed in our case study, we chose to use the paraphrazation technique (discussed in section 2.3) and reach out to a domain expert. We decided to use paraphrazation because our approach already included the generation of a natural language process description from the process models we sought to validate, therefore in validating these particular resulting artifacts of our approach, the process models were also validated. The interaction with this expert was similar to that of the interview discovery cycle (Figure 2.2): after completing the design of the process models and extracting text descriptions from them, we sent the expert all process models and their text descriptions, and an explanation of all the BPMN elements present in them. Validation of the processes was iterative. The expert sent us feedback based on their expertise in the domain, and focused on whether the processes designed reflected the real life processes they were based on, which was incorporated into the process models and reflected in the text descriptions, and these revised process models and descriptions were then sent back to the domain expert for validation. Once the expert considered all models and descriptions semantically correct, we considered the processes designed semantically validated. Semantic validation was conducted after verification of the models by a BPMN expert, as explained in the next section.

4.2 Process Model Verification

For the syntactic verification of our case study, we verified our process models with a BPMN expert. We also used the approach described in Dijkman, Dumas and Ouyang (2008) to map BPMN 2.0 process models to Petri Nets, which are then verified by specialized software in order to analyze the structure of the process models.

The BPMN expert received the process models designed and the text descriptions extracted from them, and evaluated them structurally (syntactically). After receiving feedback from this expert, we incorporated it into the process models, and these alterations were then reflected in the text descriptions of the processes. This cycle repeated itself until the BPMN expert deemed the process models structurally and pragmatically correct (section 2.3). All processes were modeled according to the 7 Process Modeling Guidelines (subsection 2.3), a requisite of our BPMN expert.

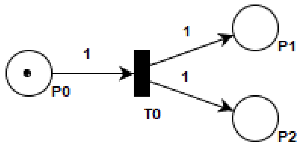
4.2.1 Petri Nets

Petri Nets are a graphical and mathematical formalism specifically designed to model systems with interacting components that can be used to describe several classes of logical, performance, and continuous and hybrid models (DAVID; ALLA, 2010). They capture many characteristics of event-driven systems, such as concurrency (the ability for tasks to be executed in arbitrary order without the outcome being affected), deadlocks (constructs that impede a process from completing), conflicts, and others (SEATZU; SILVA; SCHUPPEN, 2012). PNs are models for procedures, organizations and devices where regulated flows play a role in achieving desired results (REISIG, 1985).

A Petri Net is a directed graph with two kinds of elements: *places* and *transitions*. *Places* are passive components that can store, accumulate, or show tokens, and have discrete states. They are represented graphically by a circle or ellipse. *Transitions* model active components, and can produce tokens, consume, transport, or change them. they are represented by a square or rectangle. Transitions usually are labeled, but that label can be omitted to create "silent transitions", which represent steps that don't have impact outside of the net (DIJKMAN; DUMAS; OUYANG, 2008). An example Petri net is shown in Figure 4.1. Places and transitions are connected to each other by *arcs*, which are graphically represented by arrows. *Arcs* do not model system components, but relations between them, such as logical connections and access rights (REISIG, 2013). Arcs should never connect two places or two transitions; this is a basic property of PNs, as they are bipartite graphs Reisig (1985), Dijkman, Dumas and Ouyang (2008). If an arc is directed from an element *i* to an element *j*, then *i* can be called an *input* of *j*, and *j* can be called and *output* of *i* (PETERSON, 1977). A *Petri Net* can also be defined as a triple (P, T, F) (AALST, 1997):

- P is a finite set of *places*,
- T is a finite set of *transitions* ($P \cap T = \emptyset$),
- $F \subseteq (P\, x\, T) \cup (T\, x\, P)$ is a set of *arcs* (flow relation).

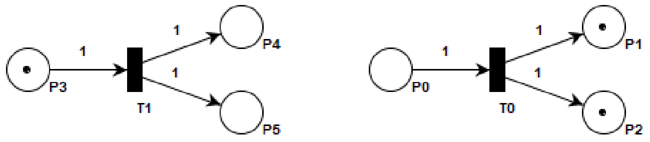
Figure 4.1: Example of a Petri net: P0 (a place) is an input to T0 (a transition), which has as outputs both P1 and P2.



Source: The authors.

In addition to these three graphical elements, *markings* are used to represent the distribution of tokens across places in the Petri Net (REISIG, 2013). They are represented graphically by a black dot inside a place (in the Petri Net). Markings represent the state of a Petri Net (DAVID; ALLA, 2010). The evolution of the state of a Petri Net, and thus of its markings, is caused by the firing of its transitions. A transition can fire only if there is at least one token in each of its inputs. When a transition fires, it consumes one token from each of its inputs, and produces one token in each of its outputs, such as in Figure 4.2 (DAVID; ALLA, 2010; DIJKMAN; DUMAS; OUYANG, 2008).

Figure 4.2: Example of a transition firing: the token on the input place (P0) to T0 is consumed, and tokens are generated in P1 and P2, its output places.



Source: The authors.

In this work, we will focus on *workflow nets*, or WF-nets, as that is the subclass of Petri Nets Woflan analyses (VERBEEK; AALST, 2000; AALST, 1997). Workflow Nets are Petri Nets that conform to the following set of restrictions: there is a unique *source place*, a single place that is not the target of any arc, a unique *sink place*, a single place that is not the source of any arc, and every other place and transition on the net is on a directed path between these two unique places (DIJKMAN; DUMAS; OUYANG, 2008). They possess a good correctness notion based on theoretical results (AALST, 1997). Workflow nets model workflow process definitions, that is, they model a life-cycle of one case of the workflow process in isolation (VERBEEK; AALST, 2000). The first restriction of WF-Nets related to how cases of workflow processes are created all at once, and to how when they are concluded they are considered completely handled, and therefore are ’deleted’; the second restriction is related to not having dangling tasks or conditions after the completion of the workflow process case (AALST, 1997).

4.2.2 Mapping BPMN onto Petri Nets

The approach taken to map BPMN 2.0 process models to PNs in this work is the one described in Dijkman, Dumas and Ouyang (2008). This approach maps a subset of core BPMN elements that deals with the order in which activities and events can occur, producing Petri Nets suitable for static analysis (DIJKMAN; DUMAS; OUYANG, 2008). Additionally, Woflan can also analyze behavioral properties from these nets. Moreover, this approach maps BPMN process models to plain Petri Nets (the fundamental definition of the formalism), which makes it easy to apply the restrictions required by the definitions of a workflow net. First the mappings employed in the approach are presented, then the BPMN process models are mapped to Petri Nets, and finally the Petri Net models are verified.

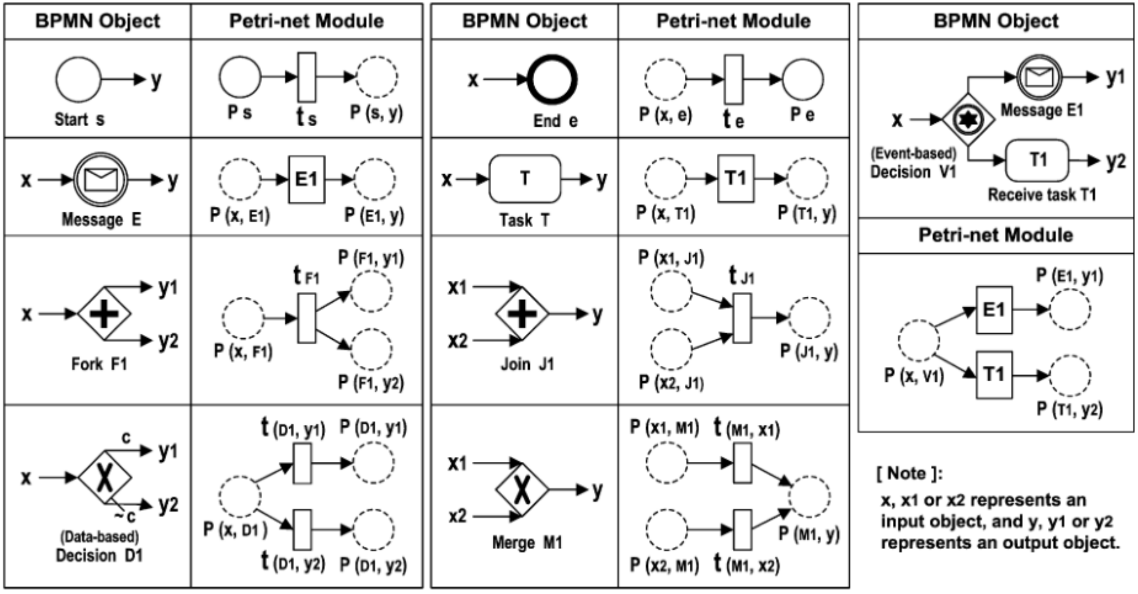
Dijkman, Dumas and Ouyang (2008) introduce the notion of "well-formed BPMN processes", which the authors define as processes with the following characteristics:

1. A start event or an exception event has just one outgoing (sequence) flow, but no incoming flow;
2. An end event has just one incoming flow, but no outgoing flow;
3. Activities and intermediate events have exactly one incoming flow and one outgoing flow;
4. Fork or decision gateways have one incoming flow and more than one outgoing flows;
5. Join or merge gateways have one outgoing flow and more than one incoming flows.

All of the process models in our case study present these characteristics, which makes this approach appropriate for our use. This happens due to the design of the process models in subsection 3.3 being in compliance to the 7 Process Modeling Guidelines (subsection 2.3). *G2: Minimize routing paths per element* is the guideline related to these characteristics.

The mapping of BPMN tasks, events and gateways to Petri Net modules used in Dijkman, Dumas and Ouyang (2008) is presented in Figure 4.3. Tasks and intermediate events are mapped onto a module composed of a transition with one input place and one output place; that transition models the execution of that task or event. Start and end events are mapped to a similar module, however such a module uses a silent transition instead of a labeled one to signal the start or end of a process.

Figure 4.3: Mapping of tasks, events and gateways from BPMN to Petri Nets.



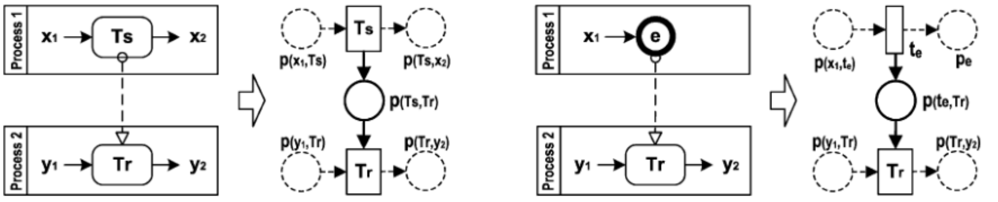
Source: Dijkman, Dumas and Ouyang (2008)

The Petri Net modules to which XOR and AND gateways are mapped use silent transitions to model their behaviour: the AND-split gateway modules employ a single silent transition, so that its firing produces tokens in all its output places and thus enables multiple paths concurrently on and AND-split and a single token on and AND-merge, while XOR gateway modules employ multiple silent transitions, so that a single token can only enable a single path. Places depicted with dashed borders in Figure 4.3 are not unique in their use to a single module, meaning they can be shared by different modules. This facilitates and simplifies the assembly of the PN model from these modules.

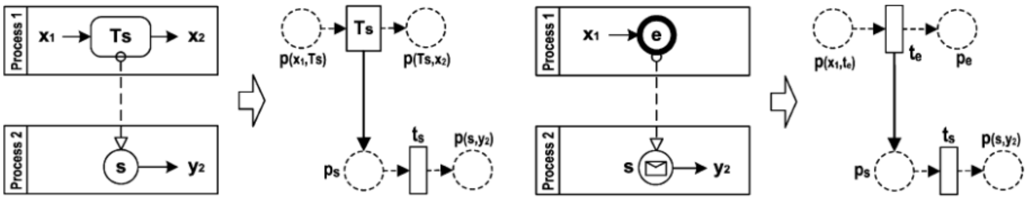
Message Flows represent interactions between different resource and participant pools (DUMAS et al., 2018). In general, they can be mapped to a place which has an arc incoming from the transition that models the sending of the message and an outgoing arc that connects to the transition that models a receive action (DIJKMAN; DUMAS; OUYANG, 2008).

There are special cases, however, that must be mapped in a different way to be correctly modeled. A message flow from a task to a start event is mapped to PNs by directly connecting the transition modeling the task to the initial place (or "trigger place") of the star event module with an arc. A message flow from an end event to a task is mapped to a place with an incoming arc from the silent transition in the end event mapping module and an outgoing arc to the transition modeling the task. A message flow from an end event to a start event is modeled by directly connecting the silent transition in the end

Figure 4.4: Mapping of message flows from BPMN to Petri Nets.



(a) task to task (b) end event to task



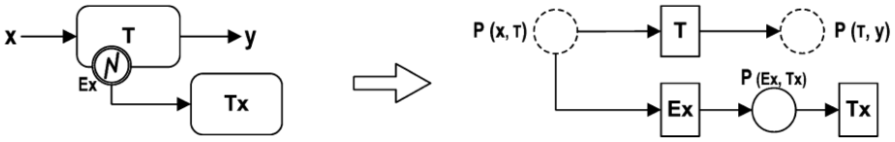
(c) task to start event (d) end event to start event

Source: Dijkman, Dumas and Ouyang (2008)

event mapping module to the trigger place of the start event mapping module. All of the message flow mappings described above are shown in Figure 4.4.

In BPMN process models, error events attached to the boundary of an activity are the origin of *exception flows*, which model exception handling in processes (DUMAS et al., 2018). For the mapping of this representation of exception handling in BPMN to Petri Nets, Dijkman, Dumas and Ouyang (2008) distinguishes between atomic tasks and subprocesses. For an atomic task, the Petri Net module that represents this is similar to that for a XOR-split, but it uses labeled transitions instead of silent transitions, as shown in Figure 4.5. The initial markings of a Petri Net model, representing the initial state of a BPMN process model, are represented by a token in the net's unique source place.

Figure 4.5: Mapping of exception handling from BPMN to Petri Nets.

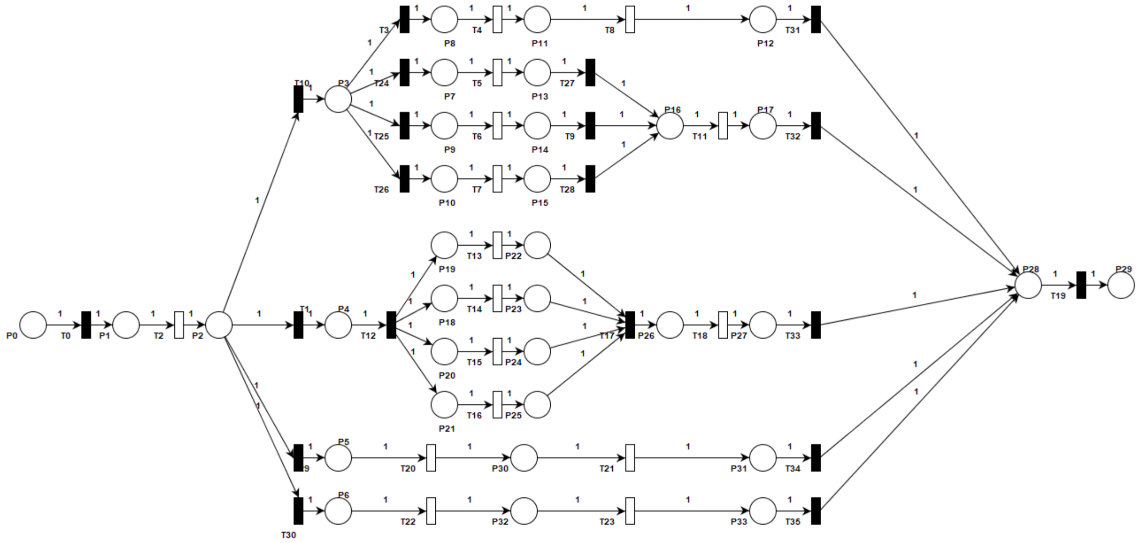


Source: Dijkman, Dumas and Ouyang (2008)

4.3 Case Study Process Models in Petri Net representation

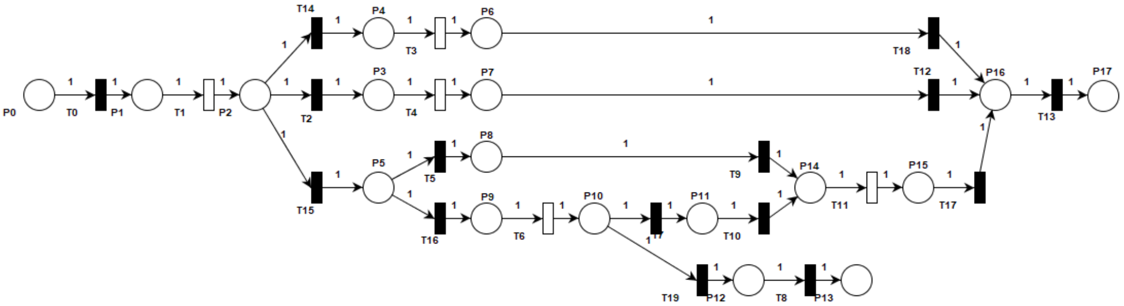
In this section we present the Petri Net representations of the BPMN process models designed in our case study, presented in Chapter 3. We applied the mapping technique described in subsection 4.2.2, using the PIPE v4.3.0 software for modeling the Petri Nets (DINGLE; KNOTTENBELT; SUTO, 2009). The reason for using this software is because it is a free software with a friendly user interface and streamlined for performance modeling. The resulting PNs are presented in Figures 4.6 through 4.9. PIPE represents places with circles, transitions with white (for labeled transitions) and black (for silent transitions) rectangles, and arcs with arrows.

Figure 4.6: Mapping of the SARS-CoV-2 Transmission process to a Petri Net.



Source: The authors.

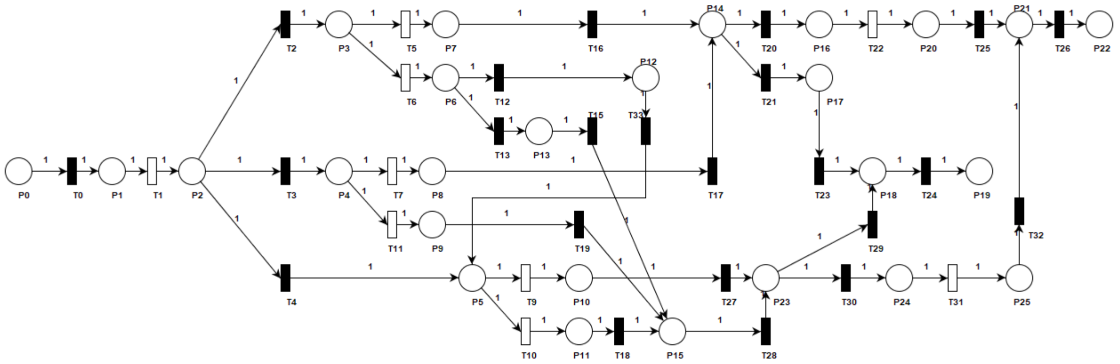
Figure 4.7: Mapping of the SARS-CoV-2 Contagion process to a Petri Net.



Source: The authors.

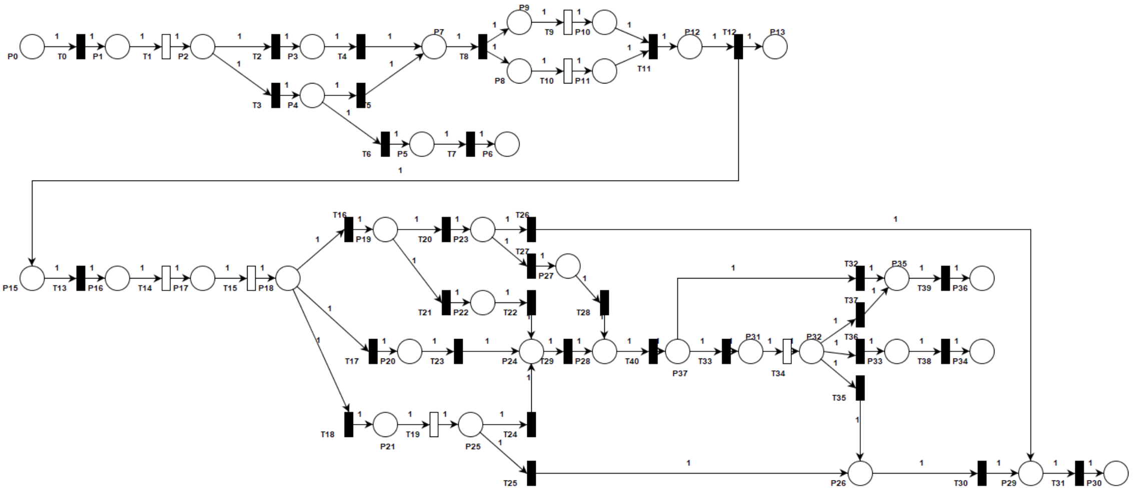
The mapped Petri Nets are visually similar to the BPMN process models in structure to the study case process models present in Chapter 3. The approach Dijkman, Dumas and Ouyang (2008) present to map BPMN process models to Petri Nets is unambiguous, intuitive, and streamlined. The mapped Petri Nets for each process allows us to verify and analyze the structure of the process models; this is presented next in this chapter.

Figure 4.8: Mapping of the COVID-19 Symptomatic Manifestation process to a Petri Net.



Source: The authors.

Figure 4.9: Mapping of the COVID-19 Identification process to a Petri Net.



Source: The authors.

4.4 Petri Net Verification

To verify the correctness of the designed process models, we used the Woflan (WOrkFLow ANalyzer) software (VERBEEK; AALST, 2000). the reason it was chosen is because it is well-documented, free, and provides valuable diagnostics and insights on Petri Nets. Woflan uses Petri-net-based analysis techniques to detect errors and verify the correctness of workflow processes. It was designed to verify process definitions (workflow nets) downloaded from workflow management systems. Woflan consists of three main parts: a *parser* than can analyze process definitions specified in terms of a Petri Net, more specifically a workflow net, and parse them into a data structure; *analysis routines* that use such data structures as a starting point for various analysis (tasks without input or output condition, detection of suspicious constructs, detection of constructs violating the free-choice property, among others) and provide information about the structure of the process definition given in the form of warnings and errors detected; and the *user interface* that allows users to interact with the software's functionalities.

Woflan is based on key concepts, two of which are *the definition of a workflow net* and *the soundness property* (AALST, 1997). Workflow nets were presented in subsection 4.2.1. The *soundness property* is related to the dynamics of the definition of the workflow process, and it is the minimal property any workflow process definition should satisfy; *soundness* also implies the absence of livelocks and deadlocks Verbeek and Aalst (2000). A workflow net is considered sound if it satisfies the following requirements:

- For any case of the workflow process, it is possible to reach a state with at least one token in the unique sink place, that is, it is possible to terminate.
- There are no tokens left behind in the workflow net at the moment the case ends (a token appears in the unique sink place), so that there are no dangling references.
- There are no "dead tasks", that is, starting from the unique source place with a token, it should be able to execute any arbitrary task following the correct route in the workflow-net.

Woflan is capable of deciding whether a given workflow net is sound or not (VERBEEK; AALST, 2000). It also warns about suspicious constructs in the definition given, such as non-free-choice constructs that often correspond to a mix of choice and synchronization, bad structuring of gateways (such as XOR-splits not being followed by XOR-joins), among others. Soundness is important in a workflow net (and in process models) because it comprehends various facets of syntactic quality (section 2.3): sound workflow nets do not have places and transitions (therefore, activities) that do not contribute to the outcome of a workflow or process, making it easier to read and understand; they do not have constructs that may cause issues in execution later on, such as deadlocks or livelocks in the workflow; they do not allow for references to an instance of workflow execution to linger after it is concluded, contributing to the organization of its executions.

Woflan expects a TPN format file, representing a Petri Net, as input to verify a process description. The PIPE software did not output Petri Nets in this file format, and the built-in file converters in Woflan did not work for the XML files PIPE output. Seeking to avoid mistakes that could come from manually converting various XML files to the Woflan-accepted TPN files, we developed a Python script that applied this conversion. This script was also extended to accept CPN files, such as those the "CPN Tools" software outputs (JENSEN; KRISTENSEN; WELLS, 2007), as we also experimented with its use, and for testing purposes. The script can be found in Appendix A and on our GitHub repository [1]. We applied the script to all our XML Petri Net files, and then used the generated TPN files with the Woflan software so that the process descriptions could be analyzed.

The Woflan software verifies a process definition's syntax by analyzing the structure of a Petri Net. An example of an error of syntax would be an arc connecting two places or two transitions. The PIPE software already does not allow for this to be done as one designs a Petri Net in it, therefore in our mapping of BPMN processes to Petri Nets, this was already an impossibility. The analysis of the files, via the data structure created by its parser, by Woflan did not point out any such errors in our models, as was expected. Suspicious constructs are fragments in the net that may cause undesirable states, such as *deadlocks* caused by complementing an OR-split with an AND-join. This kind of constructs can be avoided by following the 7 Process Modeling Guidelines Mendling, Reijers and Aalst (2010), which we present in subsection 2.3 and employed throughout our design efforts.

The Petri Net corresponding to the SARS-CoV-2 Transmission process model was verified correct by Woflan as it is presented in Figure 4.6. It was structurally analyzed and verified as a workflow process definition, all conditions were considered proper, all tasks were live (meaning there were no dead or non-live tasks) and it complied with all soundness requisites 4.4.

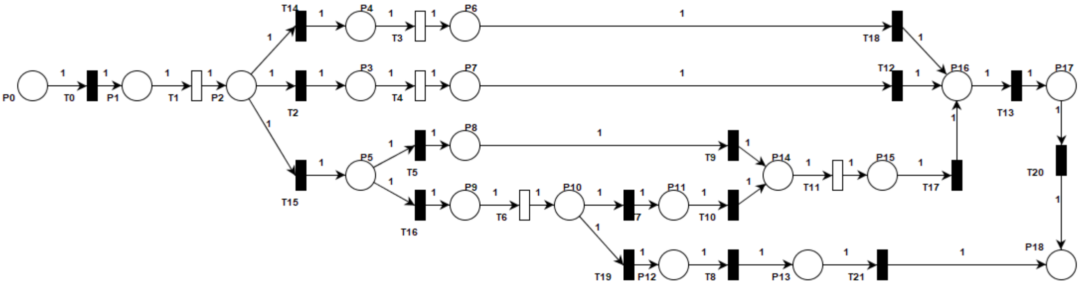
The Petri Net corresponding to the SARS-CoV-2 Contagion process model was not verified correct by Woflan as it is presented in Figure 4.7, since it had more than one end condition (or sink place), which does not comply with the definition of a workflow process description. Due to how the Woflan software is implemented, if a PN does not comply with the definition of a workflow process, it does not further analyze the process description. This error is prompted by the real-life conditions of the process that was modeled, not due to an error in design; the BPMN specification allows this (OMG, 2014). To be able to further analyze the model, we decided to make alterations to the Petri Net in order to comply with the definition of a workflow process, correcting the condition that was deemed incorrect, that is, adding transitions and a place that became the sole end condition of the Petri Net. These alterations are reflected in Figure 4.10. The alterations applied do not alter the semantics of the process or the process model, and do not

---

[1] https://github.com/Berger-DM/BPM-Pesquisa/tree/master/PIPEtoWoflan

compromise the semantic validation of our domain and BPM experts. They were applied solely on the Petri Net representation of the process model, and solely for the purpose of allowing for a deeper analysis of our model.

Figure 4.10: Mapping of the SARS-CoV-2 Contagion process to a Petri Net with a single sink place.



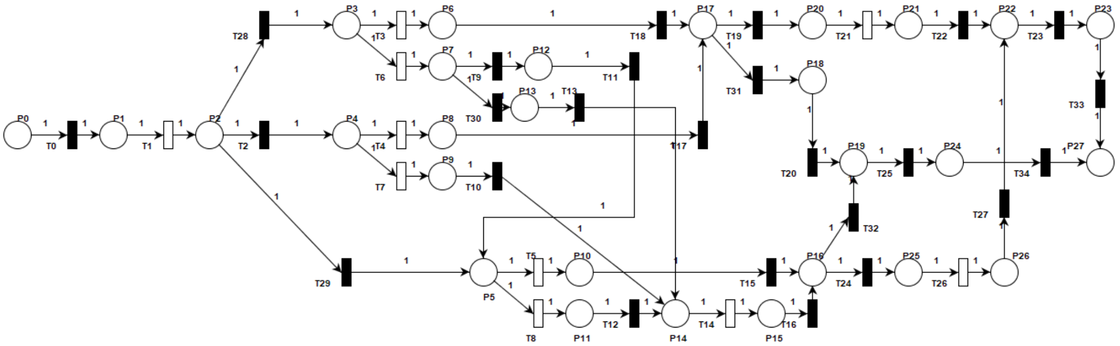
Source: The authors.

After this alteration, we ran the file conversion script on this new Petri Net and had it analyzed by Woflan. This time, the process description input to the software was verified correct, complying with a workflow process description, all conditions were considered proper, all tasks were live, and complying with all soundness requisites.

The Petri Net corresponding to the COVID-19 Symptomatic Manifestation process model was not verified correct by Woflan as it is presented in Figure 4.8, since it had more than one end condition (or sink place), which does not comply with the definition of a workflow process description. Due to this, the net representing the process could not be analyzed. This error is prompted by the real-life conditions of the process that was modeled, and such modeling is allowed in the BPMN notation (OMG, 2014). To be able to further analyze the model, we decided to make alterations to the Petri Net in order to comply with the definition of a workflow process, correcting the condition that was deemed incorrect, that is, adding transitions and a place that became the sole end condition of the Petri Net. These alterations are reflected in Figure 4.11. These alterations were applied solely to the Petri Net representation of the model for the purpose of allowing for deeper analysis, not compromising previous validation of the BPMN process model and description.

After this alteration, we ran the file conversion script on this new Petri Net and had it analyzed by Woflan. This time, the process description input to the software was verified correct, complying with a workflow process description, all conditions were considered proper, all tasks were live, and complying with all soundness requisites.

Figure 4.11: Mapping of the COVID-19 Symptomatic Manifestation process to a Petri Net with a single sink place.

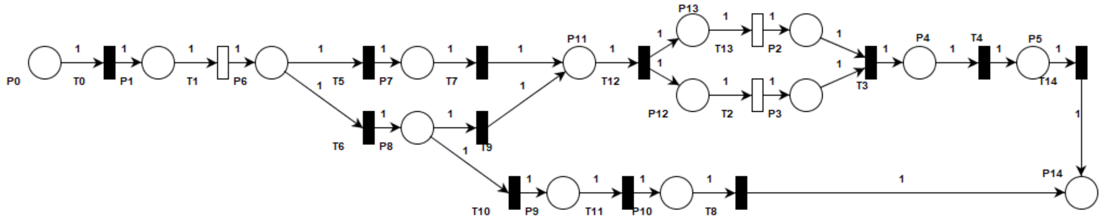


Source: The authors.

The Petri Net corresponding to the COVID-19 Identification process model was not verified correct by Woflan as it is presented in Figure 4.9, since it had more than one start condition (or source place), due to the BPMN process model having more than one pool with tasks modeled in them, and also more than one end condition (or sink place), which does not comply with the definition of a workflow process description. We attempted to alter its Petri Net mapping as had been done to the PNs corresponding to the Contagion and Symptomatic Manifestation processes, however this prompted an error in soundness, as it generated a suspicious construct in the form of an AND-OR mismatch.

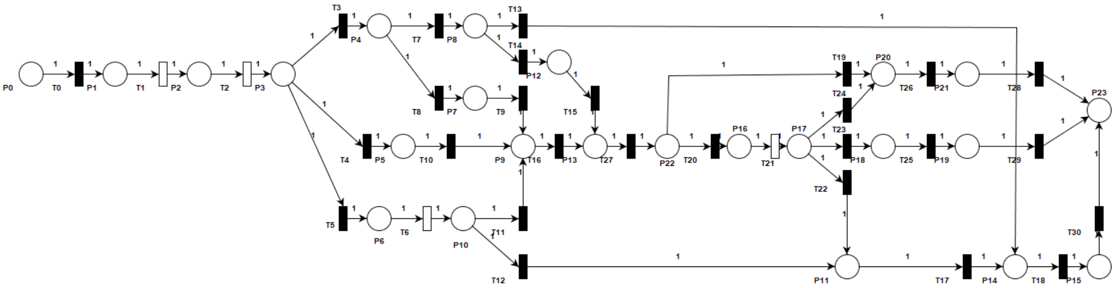
After evaluating the issue, we noticed it resided in the modeling of the message flow between the "Professional medical help sought out" end event and the "Person arrived at triage center" start event - a message flow that, unlike all the other process models analyzed, was explicitly connecting two BPMN events. All other message flows in the process models designed in this case study were between a collapsed pool (a pool that did not have any BPMN elements modeled inside it) and an event. The suspicious construct analysis in Woflan is implemented in order to help avoid deadlocks and other soundness issues (VERBEEK; AALST, 2000); however, a message flow is not the same as a sequence flow in BPMN, a distinction Petri nets are not able to map correctly. Moreover, although the approach presented by Dijkman, Dumas and Ouyang (2008) allows for the mapping of message flows, it does not comprise the mapping of organisational elements, such as lanes. For the purpose of analysis only, we chose to remove the mapping of the message flow in the Petri Net representation of the model, in order to allow for the analysis of both pools presented in Figure 3.6 separately. This does not compromise the validation of the BPMN process model by our domain expert, nor the verification by our BPM expert. Removing this particular mapping (only in the Petri net) separates the PN representing the COVID-19 Identification process into two nets: one representing the "Suspecting Person" pool, and another representing the "Medical Professional" pool. The "Suspecting Person" pool is mapped to the Petri Net shown in Figure 4.12; the "Medical Professional" pool is mapped to the Petri Net shown in Figure 4.13. Given our experience with the Contagion and Symptomatic Manifestation processes, both of these PNs are already altered to have a single sink place (or end condition).

Figure 4.12: Mapping of the COVID-19 Identification process, "Suspecting Person" pool, to a Petri Net with a single sink place.



Source: The authors.

Figure 4.13: Mapping of the COVID-19 Identification process, "Medical Professional" pool, to a Petri Net with a single sink place.



Source: The authors.

After these alterations, we ran the file conversion script on these new Petri Nets and had it analyzed by Woflan. This time, the process descriptions input to the software were verified correct, complying with a workflow process description, all conditions were considered proper, all tasks were live, and complying with all soundness requisites.

All models were verified for all properties analyzed by the Woflan software, which indicates the mapped workflow nets do not present structural defects, and were structurally verified. Therefore, the BPMN models from which these workflow nets were mapped also were structurally verified.

The errors found by Woflan in the mapped nets before the described adjustments were consequence of their real-life behaviour, in the case of the software finding it not conforming to the workflow net definition because of having multiple start and end events. The Petri Nets representation of message flows caused issues for verification on Woflan, due to generating a suspicious construct (an AND-OR mismatch). We believe this happened because Petri Net arcs could not express the difference between sequence and message flows correctly as they were proposed in this mapping approach, as they are BPMN elements that describe inter-organization communication, and Dijkman, Dumas and Ouyang (2008) did not support the mapping of organisational elements.

4.5 Chapter Summary

In this chapter, we presented the evaluation of the application of our approach to our case study: SARS-CoV-2 and COVID-19 processes. We reported their semantic validation by a domain expert, and structural and pragmatic evaluation by a BPMN expert. Furthermore, we presented their verification via Petri-net-based analysis, introducing the basic concepts of Petri Nets, an approach to mapping BPMN process models to Petri Nets (and, more specifically, workflow nets, a restricted subset of Petri Nets), which we applied to the process models designed in our case study, modeling these Petri Nets with PIPE software, and then converted the output files with a Python Script we developed in order to be able to analyze them with the Woflan software, as the file types were mismatched. We discussed the analysis results from the Woflan software for our mapped Petri Nets, and their verification as correct by the software.

5 CONCLUSION

In this work, we presented an approach for process discovery from spread out, natural language documentation. We then applied this approach to a case study, generating process models and process text descriptions for SARS-CoV-2 and COVID-19 processes. These artifacts were semantically validated with a domain expert and structurally validated by a BPM expert. Afterwards, these process models were mapped to Petri Nets, a formalism designed to model systems with interacting components and that enables complex formal analysis and diagnostics, such as the detection of dead and non-live transitions, evaluation of suspicious constructs, and soundness checks, among others. The results of these mappings were Petri Nets of the processes designed in our case study, which were then verified structurally via a Petri-net-based analysis tool, Woflan, which reported no structural flaws in process model design.

The hypotheses presented in this work were that through a process-oriented approach, it is possible to consolidate spread out documentation for the discovery of process information and process design, and that the BPMN 2.0 notation is capable of representing behavioral information of the SARS-CoV-2 virus and the COVID-19 infection; based on these hypotheses our goals were to present an approach for discovery of healthcare processes from spread-out, natural language documentation, and to generate visual and behavioral documentation of SARS-CoV-2 and COVID-19 processes. The results presented in this work corroborate those hypotheses, and we have achieved both of these goals, as presented in chapter 3.

The work developed on the case study presents SARS-CoV-2 and COVID-19 processes in a visual format, something that in our research on the literature was not prevalent. They can be used for education on subjects they represent and visualization of the disease progression and course, as we imagine healthcare professions curricula will incorporate this new virus and disease in their formation. It has also been pointed out by our domain expert that this visual representation, specially if automated, can be of aid to healthcare professionals in more remote settings, where they'd have less reference material to access.

As to limitations to our work, though our approach relied on techniques and approaches developed for use with NLP tools, all portions of our approach were applied manually, as the implementations presented and used in our related works were not available, and therefore could not be used. Moreover, each of these approaches solves a single part of the challenges faced: Friedrich, Mendling and Puhlmann (2011) propose an approach to generating process models from natural language texts; Ferreira, Thom and Fantinato (2017) propose mapping rules to identify process elements in text; Leopold, Mendling and Polyvyanyy (2014) propose an approach for the validation of process models by generating text descriptions from them, and Silva et al. (2019) sought the generation of sound process descriptions from process models. Our intent in combining portions of these approaches was to both extract process models with high accuracy to the documentation and generate text descriptions with high accuracy to the source material, which was a set of spread out documents. Data extraction and summaries were peer-checked only after the processes had already been modeled and text descriptions extracted; however, this was done by a domain expert. There might be publication bias due to having the WHO as our basis for seeking relevant data sources, however we consider this does not affect the results presented in our work as all of the sources used in this work are of renown, and due to the validation of our results by a domain expert.

Finding documentation with information for these processes' design presented a challenge, not due to lack of documents, but due to the amount of literature being published on the subject of our case study regularly. The constant evolution of the literature made redesigns necessary; as soon as this need was noticed we implemented regular, biweekly checks (and redesigns, when needed) for compliance with the most up-to-date literature. We believe this need for maintenance presents an opportunity for work to be developed on maintaining models in dynamic process domains, possibly with the aid of artificial intelligence software.

Future work is suggested towards application of our approach to other process domains, so that it may be further evaluated, and also with the implementation (or acquisition) of the tools that were replaced by manual processing of the documents, generating a comparison between manual and automatic application. Another possibility for future works is the application of this approach to other, similar, subjects, such as SARS and MERS infections, in order to generate comparisons between them. Moreover, we suggest a deeper analysis of the pragmatic quality of the models generated by our approach.

REFERENCES

AALST, W. M. P. van der. Verification of workflow nets. In: **Proceedings of the 18th International Conference on Application and Theory of Petri Nets**. Berlin, Heidelberg: Springer-Verlag, 1997. p. 407–426.

AALST, W. van der. **Process Mining: Data Science in Action**. 2nd. ed. [S.l.]: Springer Publishing Company, Incorporated, 2016.

CASCELLA, M. et al. **Features, Evaluation and Treatment Coronavirus (COVID-19)**. Treasure Island, FL: StatPearls Publishing, 2020. (Accessed: 04/05/2021). Available from Internet: <https://pubmed.ncbi.nlm.nih.gov/32150360/>.

CHEN, Y.; LIU, Q.; GUO, D. Emerging coronaviruses: Genome structure, replication, and pathogenesis. **Journal of Medical Virology**, Wiley, v. 92, n. 4, p. 418–423, 2020.

DAVID, R.; ALLA, H. **Discrete, Continuous, and Hybrid Petri Nets**. 2nd. ed. [S.l.]: Springer Publishing Company, Incorporated, 2010. 1–20 p.

DIJKMAN, R. M.; DUMAS, M.; OUYANG, C. Semantics and analysis of business process models in bpmn. **Information and Software Technology**, v. 50, n. 12, p. 1281–1294, 2008.

DINGLE, N. J.; KNOTTENBELT, W. J.; SUTO, T. Pipe2: A tool for the performance evaluation of generalised stochastic Petri Nets. **SIGMETRICS Performance Evaluation Review**, Association for Computing Machinery, New York, NY, USA, v. 36, n. 4, p. 34–39, 2009.

DU, N. et al. A case series describing the epidemiology and clinical characteristics of covid-19 infection in jilin province. **Virulence**, Taylor & Francis, v. 11, n. 1, p. 482–485, 2020.

DUMAS, M. et al. **Fundamentals of Business Process Management**. 2nd. ed. [S.l.]: Springer Publishing Company, Incorporated, 2018. 1-212 p.

FERNÁNDEZ, A. D. R.; FERNÁNDEZ, D. R.; GARCÍA, Y. S. Business process management for optimizing clinical processes: A systematic literature review. **Health Informatics Journal**, v. 26, n. 2, p. 1305–1320, 2020.

FERREIRA, R. C. B.; THOM, L. H.; FANTINATO, M. A semi-automatic approach to identify business process elements in natural language texts. In: **Proceedings of the 19th International Conference on Enterprise Information Systems**. [S.l.]: SciTePress, 2017. v. 1, p. 250–261.

FRIEDRICH, F.; MENDLING, J.; PUHLMANN, F. Process model generation from natural language text. In: MOURATIDIS, H.; ROLLAND, C. (Ed.). **Advanced Information Systems Engineering**. [S.l.]: Springer, 2011. p. 482–496.

GORBALENYA, A.; BAKER, S.; BARIC, R. The species *Severe acute respiratory syndrome-related coronavirus*: classifying 2019-nCoV and naming it SARS-CoV-2. **Nature Microbiology**, v. 5, p. 536–544, 2020.

HASSAN, S. A. et al. Coronavirus (COVID-19): A Review of Clinical Features, Diagnosis, and Treatment. **Cureus**, v. 12, n. 3, 2020.

HUANG, B. et al. Characteristics of the coronavirus disease 2019 and related therapeutic options. **Molecular Therapy: Methods & Clinical Development**, v. 18, p. 367–375, 2020.

ILAHI, L.; GHANNOUCHI, S. A.; MARTINHO, R. A real-world case scenario in business process modelling for home healthcare processes. In: **Proceedings of the International Joint Conference on Biomedical Engineering Systems and Technologies**. Setubal, PRT: SciTePress, 2016. p. 166–174.

INTERNATIONAL ORGANIZATION FOR STANDARDIZATION. **Information technology – Object Management Group Business Process Model and Notation**. Geneva, 2013. ISO/IEC 19510:2013. (Accessed: 04/05/2021). Available from Internet: <https://www.iso.org/standard/62652.html>.

JENSEN, K.; KRISTENSEN, L. M.; WELLS, L. Coloured petri nets and CPN tools for modelling and validation of concurrent systems. **International Journal on Software Tools for Technology Transfer**, Springer Science and Business Media LLC, v. 9, n. 3-4, p. 213–254, 2007.

KOPECKY, M.; TOMASKOVA, H. The business process model and notation used for the representation of alzheimer's disease patients care process. **Data**, v. 5, n. 1, 2020.

KUMAR, M. et al. A chronicle of sars-cov-2: Part-i - epidemiology, diagnosis, prognosis, transmission and treatment. **Science of The Total Environment**, v. 734, p. 139278, 2020. (Accessed: 04/05/2021). Available from Internet: <https://www.sciencedirect.com/science/article/pii/S0048969720327959>.

LEOPOLD, H.; MENDLING, J.; POLYVYANYY, A. Supporting process model validation through natural language generation. **IEEE Transactions on Software Engineering**, v. 40, n. 8, p. 818–840, 2014.

LUCIANO, B.; PINTO, A.; NUNES, S. Uso de BPMN em instituições de saúde - estudo da importância da modelagem de processos neste setor. 2020. Self-published.

LUDWIG, S.; ZARBOCK, A. Coronaviruses and SARS-CoV-2: A Brief Overview. **Anesthesia & Analgesia**, v. 131, n. 1, p. 93–96, 2020.

MAIER, H.; BICKERTON, E.; BRITTON, P. **Coronaviruses: Methods and Protocols**. 1. ed. [S.l.]: Humana Press, 2015. 1-23 p.

MENDLING, J.; REIJERS, H.; AALST, W. van der. Seven process modeling guidelines (7PMG). **Information and Software Technology**, v. 52, p. 127–136, 2010.

MINISTÉRIO DA SAÚDE. **Coronavírus**. 2021. Available from Internet: <https://www.gov.br/saude/pt-br/coronavirus>.

NIAID, R. M. L. **Transmission Electron Micrograph of SARS-CoV-2 virus particles**. 2021. Available from Internet: <https://www.flickr.com/photos/niaid/49645120251/in/album-72157712914621487>.

OMG, O. M. G. **Business Process Model and Notation (BPMN) Version 2.0.2**. [S.l.], 2014. (Accessed: 04/05/2021). Available from Internet: <https://www.omg.org/spec/BPMN/2.0.2/About-BPMN/>.

PETERSEN, K.; VAKKALANKA, S.; KUZNIARZ, L. Guidelines for conducting systematic mapping studies in software engineering: An update. **Information and Software Technology**, v. 64, p. 1 – 18, 2015.

PETERSON, J. L. Petri nets. **ACM Computing Surveys**, Association for Computing Machinery, New York, NY, USA, v. 9, n. 3, p. 223–252, 1977.

PRIEGO-ROCHE, L.-M. et al. Business process design from virtual organization intentional models. In: **24th International Conference on Advanced Information Systems Engineering**. [S.l.]: Springer, 2012. (Lecture Notes in Computer Science, v. 7328), p. 549–564.

ProMed. **PRO/AH/EDR >Undiagnosed pneumonia - China (HU): RFI**. 2020. (Accessed: 04/05/2021). Available from Internet: <https://promedmail.org/promed-post/?id=6864153>.

REICHERT, M. What bpm technology can do for healthcare process support. In: **13th Conf. on Artificial Intelligence in Medicine**. [S.l.]: Springer, 2011. (LNAI, 6747), p. 2–13.

REICHERT, M.; WEBER, B. **Enabling Flexibility in Process-Aware Information Systems**. 1. ed. [S.l.]: Springer Berlin Heidelberg, 2012.

REIJERS, H. A.; Mendling, J.; Recker, J. Business Process Quality Management. In: **Handbook on Business Process Management 1: Introduction, Methods, and Information Systems**. [S.l.]: Springer, 2010. p. 167–185.

REISIG, W. **Petri Nets**. 1. ed. [S.l.]: Springer-Verlag, 1985. (Monographs in Theoretical Computer Science, v. 4).

REISIG, W. **Understanding Petri Nets**. 1. ed. [S.l.]: Springer-Verlag, 2013.

SEATZU, C.; SILVA, M.; SCHUPPEN, J. H. van. **Control of Discrete-Event Systems: Automata and Petri Net Perspectives**. [S.l.]: Springer Publishing Company, Incorporated, 2012. 189-364 p.

SILVA, T. S. et al. A service-oriented architecture for generating sound process descriptions. In: **2019 IEEE 23rd International Enterprise Distributed Object Computing Conference**. [S.l.]: IEEE, 2019. p. 1–10.

SINGH, A. et al. Covid-19: From bench to bed side. **Diabetes & Metabolic Syndrome: Clinical Research & Reviews**, v. 14, n. 4, p. 277 – 281, 2020.

THOM, L. et al. Why rigid process management technology hampers computerized support of healthcare processes. In: **X Workshop on Medical Informatics, Workshop held in conjunction with SBC’10 Congress**. [S.l.: s.n.], 2010. p. 1522–1531.

VERBEEK, E.; AALST, W. M. P. van der. Woflan 2.0: A petri-net-based workflow diagnosis tool. In: **Proceedings of the 21st International Conference on Application and Theory of Petri Nets**. [S.l.]: Springer-Verlag, 2000. p. 475–484.

WANG, D. et al. Clinical Characteristics of 138 Hospitalized Patients With 2019 Novel Coronavirus–Infected Pneumonia in Wuhan, China. **Journal of the American Medical Association**, v. 323, n. 11, p. 1061–1069, 2020.

WANG, Y. et al. Unique epidemiological and clinical features of the emerging 2019 novel coronavirus pneumonia implicate special control measures. **Journal of Medical Virology**, Wiley, v. 92, n. 6, p. 568–576, 2020.

WESKE, M. **Business Process Management: Concepts, Languages, Architectures**. 2. ed. [S.l.]: Springer-Verlag, 2012. 3-69 p.

WORKFLOW MANAGEMENT COALITION. **WFMC-TC-1025: Process Definition Interface – XML Process Definition Language, V. 2.2**. [S.l.], 2012. 1–155 p.

WORLD HEALTH ORGANIZATION. **Summary of probable SARS cases with onset of illness from 1 November 2002 to 31 July 2003**. 2003. (Accessed: 04/05/2021). Available from Internet: <https://www.who.int/csr/sars/country/table2004_04_21/en/>.

WORLD HEALTH ORGANIZATION. **Global research on coronavirus disease (COVID-19)**. 2020. (Accessed: 04/05/2021). Available from Internet: <https://www.who.int/emergencies/diseases/novel-coronavirus-2019/global-research-on-novel-coronavirus-2019-ncov>.

WORLD HEALTH ORGANIZATION. **Global Surveillance for COVID-19 caused by human infection with COVID-19 virus: interim guidance**. 2020. (Accessed: 04/05/2021). Available from Internet: <https://apps.who.int/iris/handle/10665/331506>.

WORLD HEALTH ORGANIZATION. **Report of the WHO-China Joint Mission on Coronavirus Disease 2019 (COVID-19)**. 2020. (Accessed: 04/05/2021). Available from Internet: <https://www.who.int/publications/i/item/report-of-the-who-china-joint-mission-on-coronavirus-disease-2019-(covid-19)>.

WORLD HEALTH ORGANIZATION. **WHO Coronavirus (COVID-19) Dashboard**. 2021. (Accessed: 04/05/2021). Available from Internet: <https://covid19.who.int/>.

APPENDIX A — PYTHON XML (AND CPN) TO TPN FILE CONVERSION SCRIPT

This script receives as input one or more XML (created by the PIPE software) or CPN (created by the CPNTools software) files and converts them to TPN files, as required for input to Woflan. It extracts the places, transitions, and the relations between them, and reformats this information to the TPN file format. The script also includes a GUI for friendlier usage.

```python
from bs4 import BeautifulSoup
import PySimpleGUI as psg
import itertools
import os

flatten = itertools.chain.from_iterable

def BuildTPNFile(path, output):
    # Set filename and location for output file
    filename = path.split('/')[-1].split('.')[0] + '.tpn'
    print(filename)
    output = output.split('/')
    output.append(filename)
    final_file = os.path.sep.join(output)
    print(final_file)
    # Writing into file
    with open(final_file, 'w', encoding='utf8') as final:
        if path.endswith(".xml"):
            BuildFromXML(path, final)
        elif path.endswith(".cpn"):
            BuildFromCPN(path, final)

def BuildFromXML(path, outfile):
    places_list = list()
    transition_dict = dict()
```

```
with open(path, 'rb') as file:  # BeautifulSoup does
 ↪  not accept files with encoding that are not UTF-8,
 ↪  but rb works
    soup = BeautifulSoup(file.read(), 'xml')
    # print(soup.prettify())
    pnml = soup.find('pnml')
    net = pnml.find('net')
    # Get all places in net
    places = net("place")
    places_list = [x.get('id') for x in places]
    # Get all transitions in net
    transitions = net("transition")
    transitions_list = [x.get('id') for x in
     ↪  transitions]
    # Assemble transition information
    for transition in transitions_list:
        # Get places that send arcs to transition
        trans_ins = [x.get("source") for x in
         ↪  net("arc", {"target": transition})]
        # Get palces that receive arcs from transition
        trans_outs = [x.get("target") for x in
         ↪  net("arc", {"source": transition})]
        transition_dict[transition] = (trans_ins,
         ↪  trans_outs)
# Writing in output file
for place in places_list:
    outfile.write(f"place {place};\n")
outfile.write("\n")
for k, v in transition_dict.items():
    outfile.write(f"trans {k}\n")
    outfile.write(f"  in  {', '.join(v[0])}\n")
    outfile.write(f"  out {', '.join(v[1])};\n\n")
return
```

```
def BuildFromCPN(path, outfile):
    places_dict = dict()
    transition_dict = dict()
    with open(path, 'rb') as file:  # BeautifulSoup does
     ↪  not accept files with encoding that are not UTF-8,
     ↪  but rb works
        soup = BeautifulSoup(file.read(), 'xml')
        net =
         ↪  soup.find("workspaceElements").find("cpnet").find("page")
        places = net("place")
        places_dict = {x.get('id'): x.find("text").string
         ↪  for x in places}
        transitions = net("trans")
        transitions_list = [(x.get('id'),
         ↪  x.find("text").string) for x in transitions]
        # Assemble transition information
        for transition in transitions_list:
            trans_ins = [x.find("placeend").get("idref")
             ↪  for x in net("arc", {"orientation":
             ↪  "PtoT"})
                         if x.transend.get("idref") ==
                          ↪  transition[0]]
            trans_outs = [x.find("placeend").get("idref")
             ↪  for x in net("arc", {"orientation":
             ↪  "TtoP"})
                          if x.transend.get("idref") ==
                           ↪  transition[0]]
            transition_dict[transition[1]] =
             ↪  ([places_dict[x] for x in trans_ins],
             ↪  [places_dict[x] for x in trans_outs])
    # Writing in output file
    for k, v in places_dict.items():
        outfile.write(f"place {v};\n")
    outfile.write("\n")
```

```
    for k, v in transition_dict.items():
        outfile.write(f"trans {k}\n")
        outfile.write(f"  in  {', '.join(v[0])}\n")
        outfile.write(f"  out {', '.join(v[1])};\n\n")
    return

def startGUI():
    gui_layout = [[psg.Text('Choose files to convert:')],
                  [psg.InputText("", size=(70, 10),
                   ↪  disabled=True),
                   psg.FilesBrowse(file_types=(("XML
                    ↪  Files", "*.xml"),("CPN Files",
                    ↪  "*.cpn")))],
                  [psg.Text('Choose location for output
                   ↪  file:')],
                  [psg.InputText("", size=(70, 10),
                   ↪  disabled=True), psg.FolderBrowse()],
                  [psg.OK("Transform PIPE XML file into
                   ↪  Woflan .tpn file",
                   ↪  auto_size_button=True)]]
    window = psg.Window('PIPEtoWoflan', layout=gui_layout,
     ↪  disable_close=True)
    while True:
        event, values = window.read()
        if event in (None, 'Transform PIPE XML file into
         ↪  Woflan .tpn file'):
            if values[1] == '':
                psg.popup_ok('Output location must be
                 ↪  selected.')
            else:
                break
    pathways = values[0]
    output_location = values[1]
    pathways = pathways.split(';')
```

```
    print(pathways)
    for pathway in pathways:
        print(pathway)
        BuildTPNFile(pathway, output_location)
    gui_exit_layout = [[psg.Text("Files have been
     ↪  processed, and outputs will be at specified
     ↪  location.")],
                       [psg.Text("Click 'OK' to finish
                        ↪  execution.")],
                       [psg.OK('OK')]]
    window = psg.Window('PIPEtoWoflan - Files Processed',
     ↪  layout=gui_exit_layout,disable_close=True)
    event, values = window.read()

startGUI()
```

ATTACHMENT A — MAPPING RULES USED TO IDENTIFY PROCESS ELEMENTS IN TEXTS

Below, we present the mapping rules developed by Ferreira, Thom and Fantinato (2017) to identify process elements in texts, as described in Chapter 3 of this work.

Table A.1: Rules for Identification of Primary Activities.

Activities - primary rules		
Rules	Description	Sentence example
Rule 1	***<subject>+<verb>+<object>***	*The Support Officer* **<subject>** *updates* **<verb>** all *group calendars* **<object>**.
Rule 2	***<subject>+<aux>+<verb>+<object>*** **(in the future)**	*The secretary* **<subject>** *will* **<aux>** *send* **<verb>** to *dispatch* **<object>**.
Rule 3	***<verb>+<article>+<object>***	- *choose* **<verb>** *a* **<article>** *document* **<verb>**. - it *do* **<verb>** a **<article>** *order* **<object>**.
Rule 4	***<subject>+<verb>+<object>+ <conjunction>+<verb>+<object>***	*A client* **<subject>** *calls* **<verb>** the *help desk* **<object>** *and* **<conjunction>** *makes* **<verb>** a *request* **<object>**.
Rule 5	***<object>+<subject>+<verb>***	The *severity* **<object>** of the *claimant* **<subject>** is *evaluated* **<verb>**.
Rule 6	***<subject (occult)>+ <verb>+ <conjunction>+ <verb>+ <object>***	The first activity is to *check***<verb>** *and* **<conjunction>** *repair* **<verb>** the *hardware* **<object>**.

Table A.2: Rules for Identification of Primary Events.

Events - primary rules		
Rules	Description	Sentence example
Rule 1	***<subject>+<verb>+<object>***	*After the agent* **<subject>** has *confirmed* **<verb>** the *claim* **<object>** to the clerk.
Rule 2	***<subject>+<verb>+<agent>+ <object>***	The *SCT physical* **<subject>** file was *stored* **<verb in the past>** *by* **<agent>** the *back office* **<object>**. (passive voice)
Rule 3	***<object>+<verb in present perfect>***	...*Urgent document* **<object>** has *been received* **<verb>** by *the Manager...*
Rule 4	***<object>+<verb past>+<subject>***	a *message* **<object>** was *generated* **<verb>** to *the customer* **<subject>**.

Table A.3: Rules for Identification of Primary Exclusive Gateways (XOR).

Exclusive Gateways (XOR) - primary rules		
Rules	Description	Sentence example
Rule 1	***<verb>+ <signal word>+ <subject>+ <object>***	It first *checked* **<verb>** *whether* **<signal word>** *the claimant* **<subject>** is *insured* **<object>** by the organization.
Rule 2	***<signal word>+ <condition>+ <task/event>+ <alternative signal word>+ <task/event>***	*If* **<signal word>** *the claimant requires two or more forms* **<condition>**, *the Department of customer selects the forms* **<task>**. *Otherwise* **<alternative signal word>**, *Department of customer it requires documentation* **<task>**.
Rule 3	***<task/event>+ <signal word>+ <condition>***	After that they enter into a firm commitment to buy the stock and then offer it to the public **<task>**, *when* **<signal word>** *they still haven't found any reason not to do it* **<condition>**.
Rule 4	***<task>+<signal word>+<condition>+ <alternative signal word>+ <task>***	*The clerk checks* **<task>** *whether* **<signal word>** *the beneficiary's policy was valid at the time of the accident* **<condition>**. *If not* **<alternative signal word>**, *it send to Department of the intelligence* **<task>**.

Table A.4: Rules for Identification of Primary Parallel Gateways (AND).

Parallel Gateways (AND) - primary rules		
Rules	Description	Sentence example
Rule 1	***<task/event>+ <signal word>+ <task/event>***	*Forward the document* **<task>**. *In parallel with this* **<signal word>**, *the RCC shall also notify the Executive Board* **<task>**.
Rule 2	***<signal word>+ <task>+ <conjunction>+ <task>+ <task>***	*In parallel with this* **<signal word>**, *Department of sell send the document* **<task>** *and* **<conjunction>** *notify the department of engineering* **<task>**. Then, *the document is processed* **<task>**.
Rule 3	***<signal word>+ <task/event>***	*In the meantime* **<signal word>**, *the engineering department prepares everything for the assembling of the ordered bicycle* **<task>**.

Table A.5: Rules for Identification of Swimlanes.

Swimlanes		
Rules	Description	Sentence example
Rule 1	**The subject of the sentence**	**<subject>** perform **<task/event>**.
Rule 2	***<task>+ <indirect object>***	*She then submits an order* **<task>** to the *customer* **<indirect object>**.
Rule 3	***<event>+ <indirect object>***	*The Manager forwarded the form* **<event>** to *Official* **<indirect object>**.