Resource Policy Management using Access Control Model Enforcement in Business Processes

Paulo Sanchéz Lemos Alves

Thesis to obtain the Master of Science Degree in

Information Systems and Computer Engineering

Supervisor(s): Prof. Sérgio Luís Proença Duarte Guerreiro
Prof. Pedro Manuel Moreira Vaz Antunes de Sousa

Examination Committee

Chairperson: Prof. David Manuel Martins de Matos
Supervisor: Prof. Sérgio Luís Proença Duarte Guerreiro
Member of the Committee: Prof. José Luís Brinquete Borbinha

January 2021

Acknowledgments

I wish to thank various people for their support during this academic journey. To my supervisors Pedro Sousa and Sérgio Guerreiro for the helpful and constructive suggestions during the development of this work.

To my family, for always supporting and caring about me, especially in the moments when I was more stressed and lost. Special thanks to my parents, who had an important role in also guiding me through the entire college journey.

To my friends, specially Margarida Costa and Bernardo Furet, with whom I shared many moments during all these years in college and always felt to have their support; to Francisco Campaniço, who helped me have distractions to rest my mind during these pandemic times; and to Gabriel Pires, with whom I end up having a great friendship thanks to IST.

Abstract

Business processes are a core asset of corporations. They refer to how an organization is coordinated, to produce valuable products or services and determine the tasks and shape the work of every employee. Business Process Model and Notation (BPMN) is a standard available to model business processes with graphical representations. However, standards, such as BPMN are not thought and prepared to represent organizational policies related to the resources (i.e., people) of the organizations. This thesis proposes a BPMN framework, with the objective of having a better way for organizations to express their policies related to their resources or, informally, 'who can do what'. To capture this in business processes, access control models will be enforced to BPMN using the Atlas project as modeling tool. An access control model is a security technique to prevent unauthorized access to a system with the objective of achieving security. There are different types of access control models. In this thesis, the role-based access control (RBAC) and the attribute-based access control (ABAC) models are used. After implementing the BPMN framework in Atlas, a tool was developed for checking that every business process that uses the proposed framework is correct using a set of rules, and that will provide a system based on queries to support the actors and prevent non-compliant situations regarding the organizations' resource policies during the execution of the processes.

Keywords

Access control model, Attribute-based access control, Business process, BPMN, Role-based access control

Resumo

Os processos de negócio são um ativo principal das corporações. Referem-se a como as organizações coordenam o trabalho para poder produzir produtos e serviços. Os processos de negócio determinam as tarefas e dão forma ao trabalho de cada empregado. Business Process Model and Notation (BPMN) é uma norma existente para modelar processos de negócio usando representações gráficas. Contudo, normas como o BPMN não estão pensadas e preparadas para representar políticas das organizações relacionadas com recursos (i.e., pessoas). Esta tese propõe uma BPMN framework, com o objetivo de ter uma melhor forma de expressar as políticas relacionadas com os seus recursos, ou dito de outra forma 'quem pode fazer o que'. Para que isto seja capturado dentro dos processos, modelos de controlos de acesso serão aplicados ao BPMN. Para isso, a ferramenta Atlas será usada. Um modelo de controlo de acesso é uma técnica de segurança, usada com o objetivo de obter segurança e evitar acessos não autorizados a sistemas. Nesta tese serão usados o Controlo de Acesso Baseado em Cargos e o Controlo de Acesso Baseado em Atributos. Depois da implementação da framework no Atlas, uma ferramenta será desenvolvida para verificar os processos que usam a framework proposta e que proporcione um sistema baseado em queries que evite situações de não conformidade no processo por parte dos atores em relação às políticas das empresas relacionadas com os recursos durante a execução dos processos.

Palavras Chave

Modelos de Controlo de Acesso, Controlo de Acesso baseado em Atributos, Processos de Negócio, BPMN, Controlo de Acesso baseado em Cargos

Contents

List of Figures

Listings

Acronyms

ABAC	Attribute-based Access Control
BPMN	Business Process Model and Notation
GDPR	General Data Protection Regulation
RBAC	Role-based Access Control
SWRL	Semantic Web Rule Language
XACML	Extensible Access Control Markup Language
WSACML	Web Services Access Control Markup Language

1

Introduction

Contents

Authorization is present in every form of information technology and is concerned with how users can access resources in computer systems or, informally speaking, with ' who can do what' [1]. We consider access control models as a way to offer the guarantee that only qualified users can gain access to what was assigned to them. Today, business process models (BPM) [2] are the core elements of an organization and refer to how an organization is coordinated and how its work is organized to produce valuable products or services. Business processes are defined as a collection of inter-related events, activities, and decision points that involve several actors [2].

There are different standards to express the business process of an organization, being one of the most used the Business Process Model and Notation (BPMN) [3]. Withal, such standards are not though to represent authorization constraints [4]. Integrating the domains of access control models and business process models empowers the enforcement of business process operation control [3], in specific, the business process compliance.

Business process compliance is the operation that centers on asserting the business processes are compliant to the regulations, standards and internal policies of an organization [5]. The non-compliance of the business processes can be considered a threat to the organization since it can damage the production of valuables (products and services). For instance, by making them more expensive to produce. Business process compliance normally demands an event-based behavior modeling that orchestrates the communication between actors and used resources (e.g., information, time). However, most of the business processes compliance solutions consider that ex-dure is fully known, which does not always happens, and leads to organizations having an incomplete vision of the process [6].

To contribute with a solution for this problem, we propose the enforcement of an authorization approach based on access control models: integrate Attribute-based Access Control (ABAC) [7], and Role-based Access Control (RBAC) with the specification of the BPMN [8], this way creating an expansion of the standard. This will allow the specification of organization policies related to the 'who can do what' within the organizations' business process models; and the prevention of the non-compliance behavior by the actors involved in the tasks of the business processes.

1.1 Objectives

The organizational activity can be divided into three intervals in time: the ex-ante, the ex-dure, and the ex-post. Each of these intervals focuses on a particular phase of the business processes. The ex-ante is centered on what happens before the process starts functioning (i.e., this interval centers on the process modeling). This phase enables a common understanding and analysis of the business process. Hence, it is important to model correctly the business processes [9]. The next interval is the ex-dure and occurs during the execution of the business process. This interval has the objective of supporting the operation directly from the ex-ante model's definition. During this interval is where the mistakes and non-compliance can occur if the process is not clear and understood by the actors. Lastly, the ex-post focuses on what is going to happen after the execution of the process. In other words, the goal of the ex-post interval is to estimate the future behavior of the process from the available data from the past executions [10].

This thesis centers on the ex-ante and ex-dure interval. The main objective is to solve the problem of representing and operating exactly what was assigned to the actors of the organizations, to avoid non-compliance situations of the business processes and produce damage to the production of products and services of the organizations.

Example: Have an adaptation of business process Order Fulfillment[1] from [2] as an example (figure 1.1). The business process is carried out by a Seller's organization, which has a Sales department and a Warehouse & Distribution department. The process starts with a purchase order received by the Warehouse & Distribution department, where it is checked if there is stock available for the product ordered. If the product is in stock, it is retrieved from the Warehouse & Distribution before Sales confirms the order. Afterward, the Sales department creates and sends the invoice and waits for the payment, while in the Warehouse & Distribution department, the product is being shipped. The process ends with the insertion of the order in the Orders database by the Sales department. On the other hand, if the product is not in stock, the Warehouse & Distribution checks the availability of the raw materials by accessing the Suppliers Catalog database. Once the raw materials have been requested and obtained, a Warehouse Worker manufactures the product, and equally to the other branch, the product is retrieved from the Warehouse & Distribution before Sales confirms the order.

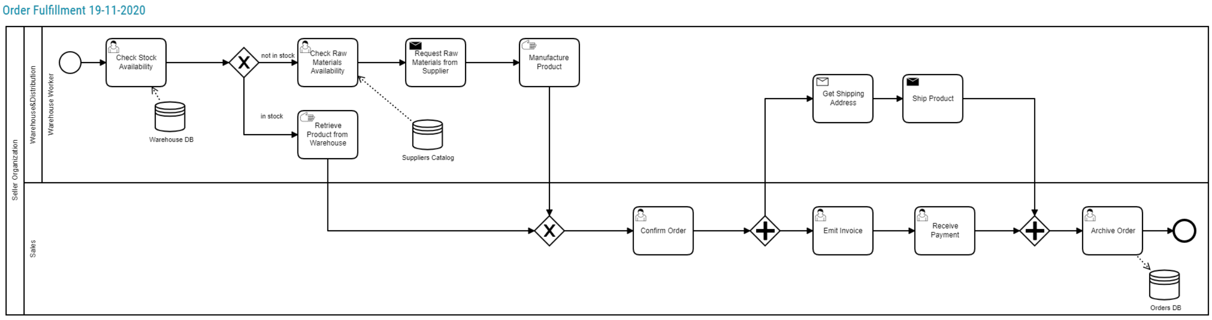


Figure 1.1: Order Fulfillment business process

By having the business process executed wrongly, and the actors not complying with the Seller's Organization policies that establish what was assigned to them, problems can occur, such as receive

[1] Since this thesis focus on lanes that represent actors rather than systems. The ERP lane presented in [2] was withdrawal and the process was customized to better fit the problem that we aim to solve

customers' complaints or have the company spend more money than the needed for executing the process. For instance, these cases can happen if the product manufacturer does not follow the specifications given the customer's order, or if the task for checking for the stock was not done, causing to buy unnecessary materials and the manufacture of a product that was available in stock.

Therefore, the contributions of this work are the following:

- Propose a solution for representing resource policies in business processes.
- Integrate access control models in the BPMN meta-model.
- Implement the previous point in the Atlas project, a Link Consulting's tool[2] for modeling. However, because of Atlas not allowing inheritance relations between classes, an interpretation of the meta-model needs to be done first.
- Develop a tool for checking the correctness of business process and giving a system based on queries, that prevents the actors of executing business process instances that are non-compliant by using the concept *need-to-know* (i.e., allow actors to only be able to access the information that is his relevant to them).
- Apply the Rent-a-Car case study's business process to the presented solution to solve the problems that the organization is having based on the ex-post information; and show the obtained results.

Hence, the first part of the solution is related to the ex-ante interval since it refers to the modeling of the business processes, and the second part centers on the ex-dure, since the tool aims to support the actors during the execution. So, while the modeling of the business processes with access control models provides the distribution of the process' activities within the actors of the process and creates the set of logically related tasks and behaviors that organizations develop over time to produce specific business results [11]; the tool with the query system in combination with the diagram helps the actors comply to the process, and maintain a good process behavior, by showing them only the tasks that they what was assign to them in a given business process.

1.2 Work Methodology

To create the solution, the followed method was based on the development of the framework and the tool followed by an assessment of what was intended to be implemented by this work. The process began with the problem identification, and research on the topics treated in this work. Then, it was

[2] www.linkconsulting.com/atlas/

followed by the cycle of implementation, and assessment based on the concepts that the related work implemented and the suggestions given by the supervisors. This cycle was repeated throughout the entire development of the work since weaknesses were found in the solution during the construction of the framework and the tool (see section 4.2.2). Lastly, once the solution had all the intended properties, it was applied to a scenario to solve the problems that this had.

1.3 Organization of the Document

This document is organized as follows. It starts with Chapter 2, presenting definitions of terms that will appear throughout the document. Then, Chapter 3 presents the conceptual background. Subsequently, Chapter 4 designs the solution to integrate RBAC and ABAC with BPMN, and the tool for checking business process along with the query system. Afterwards, Chapter 5 presents a case study with problems in terms of a business process, and the results of applying the solution to the case study are shown. Finally, Chapter 6 concludes the thesis, reflecting on the solution, and points to future work.

2

Background

With the objective of better understanding the topic of this dissertation, some relevant concepts must be first explained. This section aims to introduce these terms that will appear throughout the entire document, by explaining them.

- **Access Control Model:** Is a security technique to prevent unauthorized access to a system or resource, intending to achieve confidentiality [12]. Access controls can be either physical, to limit the access to places, or logical, to limit to digital content[1]. Normally, access control models are based on credential techniques, that identify the individual that is trying to access the resource (e.g., the login page of a Facebook account). These credentials can include passwords, PINs, security tokens, or even bio-metric scans (e.g., fingerprints). There are different access control models, depending on the controls and restrictions that the organizations what to make. In this work, only two will be used. These are the RBAC and ABAC.

- **Role Based Access Control (RBAC):** Is an access control method based on roles, where requests to perform operations on objects are granted, or denied, based on the role of the user asking for them. ”The basic RBAC defines several roles, which typically represent organizational positions such as secretary, manager, and employee” [13]. Although a very simple model, RBAC, can significantly simplify access control management for large numbers of users because it allocates permissions to roles rather than individuals [13] [14]. This simplification has led to the widespread adoption of RBAC as the access control. However, many organizations face growing diversity in terms of users and the access needs [14] and require a more granular specification of their policies. This can be solved by using ABAC.

- **Attribute Based Access Control (ABAC):** Is an access control method where the subject's requests to perform operations on objects are granted based on assigned attributes of the subject, object, and environment conditions [15]. Therefore, ABAC is a more complete access control model that allows organizations to express in a more specific manner their access policies. As organizations tend to keep up with leading-edge technology solutions, they face the challenge of managing identities, and the access of these, to a diverse number of applications and resources [14]. ABAC gives a solution for these challenges by being an advanced method for ”managing access rights for people” and by offering a greater level of flexibility and granularity than the traditional access control methods [14], such as RBAC.

- **Business Process Model and Notation (BPMN):** ”Business processes represent a core asset of corporations” [2] and they refer to how every organization is coordinated and how work is organized to produce valuable products or services. ”They determine tasks, jobs, and responsibilities and

[1] `searchsecurity.techtarget.com/definition/access-control`, Accessed= 15/01/2021

by this, shape the work of every employee” [2]. A well-known standard for modeling business processes is the BPMN that has the objective of making it easier to work with business processes [16]. ”BPMN is proposed as a modeling language for the intrinsic details of the state machine of a business process, rather than for other business dimensions” [3]. Business processes are composed of activities, events, and gateways. ”Events correspond to things that happen atomically, meaning that they have no duration” [2]. Gateways allow or disallow the passage of tokens. These type of business process elements have the objective of controlling the flow of the process through sequence flows [8]. This work expands the BPMN language proposing a new meta-model where additional information about access control models is integrated. Before presenting the changed meta-model, an extract of the original model [8] is represented in figure 2.1. The figure was created based on the information available in the articles: [4], and [8].

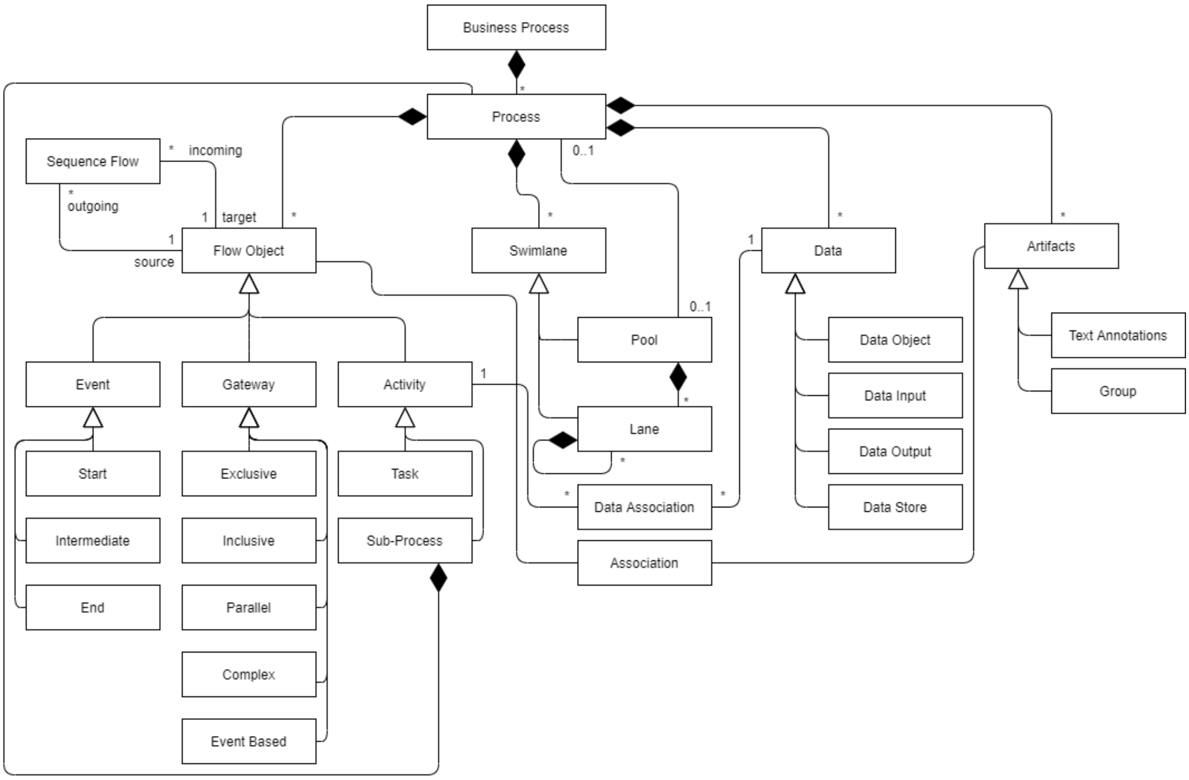


Figure 2.1: Extract of BPMN meta-model

- **Pool:** is a container for partitioning a process from other pools/participants. In this work pools are considered more to be departments or organization that contain the different organizational roles, rather than being directly a role itself. Pools can be black box or white box. A black box pool represents an external process where details are hidden [8]. A white box pool, contains visual representation of the process, i.e., activities within the pool are organized by sequence Flows and message Flows [8]. Pools can have sub-partitions, that receive the name of lanes, where the internal roles, or departments, of the organizations are represented.

- **Lane:** Is a partition used to organize, and categorize activities within a Pool [8]. Commonly, a lane represents things such as internal roles (e.g., Manager, Associate), systems (e.g., an enterprise application), or an internal department (e.g., shipping, finance) [8]. Lanes can contain other lanes inside them. These are called nested-lanes, and they are used for having a more specific categorization of the activities represented in the business processes.

- **Activity:** Is the unit of work that ”a company or organization performs using business processes” [8] to produce valuables. When an activity is rather simple and can be seen as one single unit of work, it receives the name of Task. Otherwise, if the activity is complex and can be decomposed they are Sub-Processes. Activities are process steps that can be either manual if executed by a human or automated if executed by a system [8]. In the meta-model, activities are associated with data elements through Data Association. These are dotted lines with arrowheads, and indicate the flow taken by the information [8]. If the arrowhead points to the activity, it means that the information is being used as input. Otherwise, the information is an output.

- **Data elements:** these are, Data Object, Data Input, Data Output, and Data Stores. ”Data Objects provide information about what activities require to be performed and/or what they produce” [8]. To differentiate if a data object provides information or is the product of an activity, data associations are used as explained in the previous bullet. Data inputs and data outputs provide the same information for Processes as data objects [8], but they are represented containing a small block arrow. If the arrow if unfilled, then it represents a data input. If the arrow is filled, it is a data output. Because the Atlas tool for modeling does not support these types of elements (i.e., Data Input and Data Output) they were not taken into consideration when creating the solution. Finally, data stores provide a mechanism for activities to retrieve or update stored information that will persist beyond the scope of the process [8].

- **Business Function:** In this work, is seen as an internal perspective and is a lower level activity to which there is a privilege associated. This term was created and introduced to the solution because the activities' elements of BPMN do not have any association with security requirements, such as access controls. Hence, it was necessary to have something that created this relation. Yet, because adding a new graphical element to BPMN could cause changes in the representation and sequence flows of BPMN, the term was integrated into the activities.

- **Business Process Compliance:** ”Processes count to the most important assets of companies” [17]. The non-compliance of the business processes can be considered a threat to the organization since it can damage the production of valuables (products and services). For instance, by making them more expensive to produce or by making customers unhappy and leaving complaints. Ensuring the compliance of processes to regulations, governance guidelines, strategic business requirements, and business policies is a condition without there can not achieve the control of the business behavior [17].

There are different approaches to business process compliance. All seek to discover methods and techniques to ease the implementation of the regulations [18]. Two of the approaches are the methods for validating and verifying the processes, which belong to what [18] calls *Formality* dimension and dominate the business process compliance research. The validation of processes aims to check if the end-to-end process functions as it should. That is, the validation of a process will verify if, for a given variety of inputs, the process instances are working as intended, and they produced the same output. Therefore, validation could be seen as an analysis of data [18]. Meanwhile, verification has the goal of seeing if what is documented as a property is a specification of the system [18].

In [19], [17], and [20], they defend that there are two approaches of checking for compliance, being them *forward checking* (related to verification) and *backward checking* (related to validation).

– *Forward Checking* targets the verification of rules during the design time and execution. Hence, these techniques aim to prevent the non-compliance behaviors of the processes [20]. This type of approach takes place during the design and run-time of the business process life-cycle, meaning that during the design phase, the work will be center on ensuring that process instances will be regulatory compliant [18]. On the other hand, during run-time, the objective is to gain additional information if compliance violations are observed to ”improve the underlying architecture and mechanisms for compliance checking” [18].

– *Backward Checking* can detect the non-compliance behavior by looking at the history of the business process instance's execution [20]. However, backward checking techniques' main flaw is that they can neither prevent the occurrence of non-compliant situations nor modify the behavior of the process instance during its execution to solve problems [19]. This because these techniques only compare the results obtained, by the execution of the instances, with the expected behavior [19], that is, they calculate a deviation.

When checking for compliance, another important aspect to keep in mind is what is intended to be check. In [19], the authors present different aspects being the most relevant for this work, the *Resources*. People interacting and participating in the execution of the process are considered resources. Rules stating how they must interact with the process may be defined [19], as well as who can do particular tasks. This thesis gives a compliance solution in which the organization's policies, related to the resources, are applied and integrated using access control models with BPMN control flow diagrams, ensuring that every part of the process is assigned to somebody, and intending to prevent non-compliance during the run-time of the process.

- **Separation of Duty (SoD):** Is a constraint that makes more than one subject to be required to successfully complete a process [21]. The constraint has been studied for a long time and accepted as a fundamental approach to prevent fraud and privileges misuse and abuse [22]. This is a constraint thought to prevent a singular person able to execute all critical tasks in a workflow, so that no fraud is committed [23].

- **Binding of Duty (BoD):** Is a constraint that requires that if a certain user executed a particular task then, that user, must also execute a second task or more in the workflow [21], [24]. Equally to SoD, this is a constraint to enforce a secure workflow. However, using both of these constraints should not prevent the workflow from being completed. Given a set of constraints, and sets of authorized users, there should exist a workflow execution that satisfies all the constraints [24].

- **Need to Know:** A subject should only be able to access the information that is his responsibility and is strictly necessary to carry out tasks that are his responsibility [21] [25]. This concept is implemented in one of the queries developed in the tool. To access a process in the tool first the login must be done using the Atlas's credentials. Once logged and with a process loaded in the tool, the query uses the information stored in Atlas and crosses it with the information available in the business process when this was modeled, and presents merely the activities and business functions that are the responsibility of the user logged in.

3

Related Work

In this chapter, a review of the literature on the subject is presented. It is centered on topics such as Business Process Compliance, Business Process, and Access Control Model Integration, giving special attention to the Role-based access control Model and Attribute-based access control model because they are the models used in this work.

Modeling business processes correctly is a fundamental task since it's the center for conducting and improving how the business operates [26]. Security requirements, such as authorization constraints is also a relevant task concerning the performance of the processes when running. However, many times thought after the modeling [26]. ”Access Control ensures that only the intended people can access security classified data and that these intended users are only given the level of access required to accomplish their tasks” [27].

For the creation of this Related Work's chapter, papers about the combinations of the topics of business processes, access control models, business process compliance, and BPMN were searched in Google Scholar[1], in all the AIS eLibrary's repositories[2], and ACM Digital Library[3], and the most relevant results, for this work, were selected. The papers present meta-models adapted for an expansion of the BPMN, or UML2 language, to integrate access control models among other things. Here we briefly summarize the most related work.

In [4], the authors propose an improvement of the Business Process Modeling Notation to specify authorization constraints, such as role-task assignments, role hierarchies, separation of duty and binding of duty constraints, without affecting the control flow semantics of BPMN [4]. The authors chose the BPMN language because of the expressiveness and the extensible capabilities that allow extending the language without affecting and changing the footprint. The authors, then, present the refinement of the Business Process Modeling by indicating five elements that they consider necessary to express authorization constraints. These are *Lane* and *Nested Lane*, *Manual Task*, and *Group*, which are elements that already exist in BPMN; and *Authorization Constraint Artifact*, which the authors derive from the BPMN's *Text Annotation*. They justify these terms by explaining each of them and saying that ”an activity is a generic term for a task that someone performs” [4] and *Lanes* are where the tasks are assigned to organizational roles and refer to ”the classical role-task authorization” [4]. *Nested Lanes* are used to ”represent the role-based task authorization inheritance and role hierarchy” [4]. That is, if there is a lane that contains sub-lanes, the lane inherits the task's authorizations of the sub-lanes. *Groups* are ”visual mechanism to group elements of a diagram informally” [8]. They do not affect the sequence flow of the process and can be used for documentation purposes [4] and for assigning dedicated authorization constraints to activities' groups. Lastly, *Authorization Constraint Artifact* give semantic to *Groups*. They

[1] scholar.google.com/
[2] aisel.aisnet.org/
[3] dl.acm.org/

indicate if there is a binding or separation of duty in the activities contained in the group by specifying how many users can perform the tasks of the group and how many tasks they each can allocate. An example of the solution presented in the paper can be seen in 3.1. The lanes indicate the roles authorized to execute the tasks. In the case of A1 and A2 because of the role hierarchy they can be performed by either a Manager or a Clerk; on the other hand A3 can only be perform by the Manager. The grouping of A1 and A2 with the *Authorization Constraint Artifact* can indicate either a separation or a biding of duty. The first number of the tuple presented indicates the number of users that have to perform at least one task and the second number indicates the number of tasks that each user is allowed to perform. In this case the *Authorization Constraint Artifact* indicates a separation of duty between the A1 and A2.

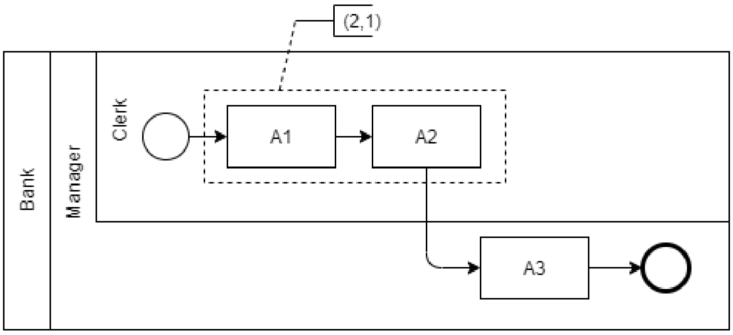


Figure 3.1: Example of C.Wolter et al. 2007 solution

In [21], the paper presents a tool that supports both design-time modeling and run-time in the enforcement of security requirements for business process-driven systems. The idea of this paper is that many times software development methods ”treat non-functional requirements, such as security, separately” [21]. However, the process behavior and the security requirements (e.g., access control, separation of duties, binding of duties, need to know) are not independent of each other, and having this separation makes it difficult to ensure that the system fulfills the requirements [21]. With this, the paper presents a tool for modeling these requirements into the business process using BPMN. The tool differentiates between two types. Ones too complex, that are represented in a diagrammatic BPMN extension, and others that are an extension of the user interface. In the first kind, there are the requirements of separation of duty, and binding of duty, while in the second kind, there is the specification of the access control models (i.e., roles and their permissions). In the case of the paper, RBAC is the only available.

In [26], the authors present an extension for BPMN for modeling security requirements. One main problem pointed out in the paper was that often the notion of security is neglected in business process models, which usually concentrates on modeling the process in a functional manner [26]. Hence, frequently security requirements are considered after the definition, which can lead to vulnerabilities [26]. The authors defend the idea that most engineers in charge of defining, documenting, and maintaining the requirements, are not trained in security, and if they are, it's an overview of security architectural mechanisms, such as passwords and encryption. With this in mind, the authors present a BPMN extension in which business processes are modeled with security requirements in a graphical way, increasing the scope of expressiveness [26]. The authors chose BPMN as the language to use because it supports visual concepts that allow the representation of the security requirements. The proposal consists of having associated a symbol (padlock) to represent security requirements in a standard way [26]. Each requirement has a special padlock with a capital letter in the center, which indicates the type (e.g., access control is represented by a padlock with the initials "AC"). The paper focuses more on aspects related to security requirements, such as non-repudiation or attack harm detection, which gets away from the scope of this thesis. However, the paper was selected as relevant since it presents another solution for modeling access control models using BPMN. Since the paper treats access control in a general manner and does no specify any specific model, figure 3.5 classifies the paper as not specific.

In [28], they arise the problem of requirements in the area of identity management often being specified in a non-formalized manner, as unstructured text by the business department. Identity Management is responsible for ensuring the quality of identity information such as identifiers, credentials, and attributes and using it for authentication and authorization [29], with the purpose of users accessing appropriate data [30]. A separation between the business process model and its related identity management requirements may easily result in inconsistencies [28]. In the paper, the authors proposed a model-driven solution for service-oriented architectures by creating a meta-model, using UML2, for "modeling access control requirements at the business process level" [28]. The model proposed in by the authors consists in that every IdMRole (in terms of BPMN, Lane) has one policy associated. A policy may be associate either with a IdMAction (in terms of BPMN, Activity) or with a IdMActivityGroup (in terms of BPMN, Group) and aggregates Permissions, that are what defines the 'who can access what'. The Permissions has at least to aggregate one Assertion, that among other things, contains the attributes regarding the subject (business role), and the object being accessed, and are in charge of allowing access on resources. From the Permission is derive DraftedPermission that allow to add comments ("access control statements without formalisation" [28]) in case a it wasn't able to define a properly policy with the available modeling concepts. Once the business process are modeled using the meta-model a XML can be extracted from them with security information, for later a Java-based application to parse them to Web Services Access Control Markup Language (WSACML) policies for a certain product. Figure 3.2 shows a simplification of an example given by the authors. The example is a banking process, where the IdMRole "Account Manager" has to do the sequence of IdMActions: "Create General Contract", "Check Account Opening", "Call Credit Rating Service", and "Call Scoring Service". These tasks are put together in an IdMActivityGroup and have a Policy associated. "The policy limits the roles that are allowed to execute those actions to the role 'Account manager'" [28], and has associated contextual restrictions, for instance 'Opening a current account is allowed only at the office times from 7 am to 7 pm.

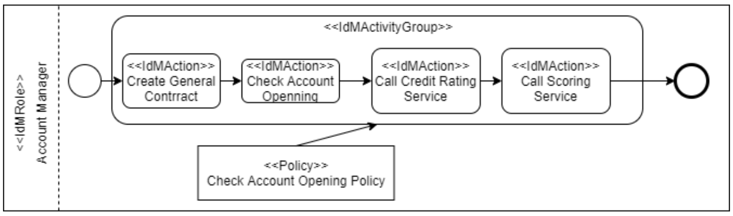


Figure 3.2: Example of H.Klarl et al. solution

In [31] the proposed solution integrates the security requirements *Separation of duty*, *Binding of duty*, and *access control* into business processes by expanding BPMN by adding graphical elements to the language. The authors expand BPMN with three types of authorization requirements, being them: multi-level security, where the authors use the Bell-La Padula models; role-based authorization requirement, where the authors use RBAC model and role hierarchies; and Attribute-based authorisation requirement, that the authors use to implement the requirements of separation and binding of duty. To model RBAC requirements the expansion uses graphical elements put on top of either lanes or activities. In case that the lane has a parent-lane the role inheritance will be implemented, however, this can be stop by adding a graphical element to the activities that should not be done by a higher role. If added the same role graphical element to an activity, it can be done a specification of the role to a person. In relation to multi-level security, the authors include to different types of visual elements that are applied to lanes or data objects, one representing clearance level (e.g., internal, external, public) that follows the Bell-La Padula model, meaning that if a lane has a certain level of clearance in can only write data for that level or above (No-Write-Down), and can read data for that level or bellow (No-Read-Up); and another representing the integrity level (e.g., top-secret, secret, high, low) and follow the Biba model. Lastly, concerning attribute-based authorisation requirement, these can be applied to activities and data objects, and are use to implement requirements such as separation and binding of duty, indicating a condition between to activities. Figure 3.3, shows a simple example of this expansion using only RBAC and ABAC. The visual elements in Manager and Clerk indicate the roles and there is a role inheritance from Manager to Clerk. The other visual elements represent ABAC requirements in the expansion. By putting them on top of A1 and A2 they are indicating a condition, in this case the condition is that the performer of the task A1 must be different from the performer of A2, implementing this way a separation of duty.

In [32] the authors give a solution for modeling privacy-aware business process. To do that, they expand BPMN to "incorporate visual constructs for modeling privacy requirements". These requirements are access control, separation and binding of duties, user consent and necessity-to-know. These elements have a visual representation in the models, in terms of access controls, these are mapped to pools and

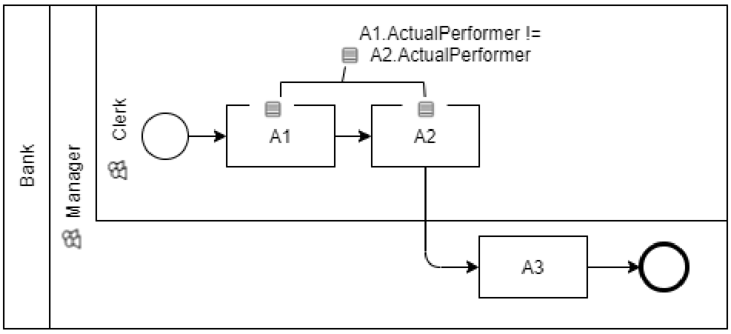


Figure 3.3: Example of C.Wolter et al. 2010 solution

lanes and the authors differentiate three types of accesses: ”Allow”, ”Prevent” and ”Limited”. Similarly, the other properties also are represented by visual elements and, apart from access control, only the property of necessity-to-know differentiates three levels. These are: ”High”, ”Medium”, and ”Low”, and the complete data is only given if there is a level high of necessity. The authors use Semantic Web Rule Language (SWRL) to represent each of the visual elements added to the BPMN. Figure 3.4, shows a simplification of the example given by the authors in the paper, and corresponds to a fire emergency situation handling at Airports. In the figure only the access control property is represented. The blue icon represent an access control ”Limited” and the SWRL representation for the actions of the main actor of this scenario is:

```
"user(martin) ∧ hasRole(martin,airportPersonnel) ∧ Resource(allairpotResoures) ∧
Action(emergencyResponse) → GrantAccess(martin, allairpotResoures)"
```

Which grants access to the user martin to the resources: ”allairpotResoures”.

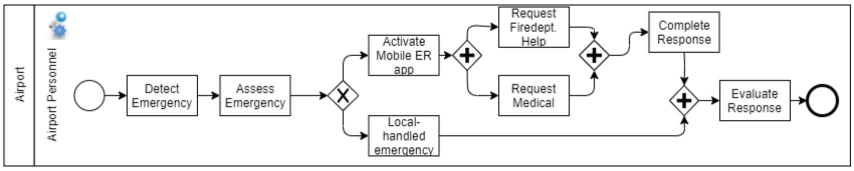


Figure 3.4: Example of W.Labda et al. 2014 solution

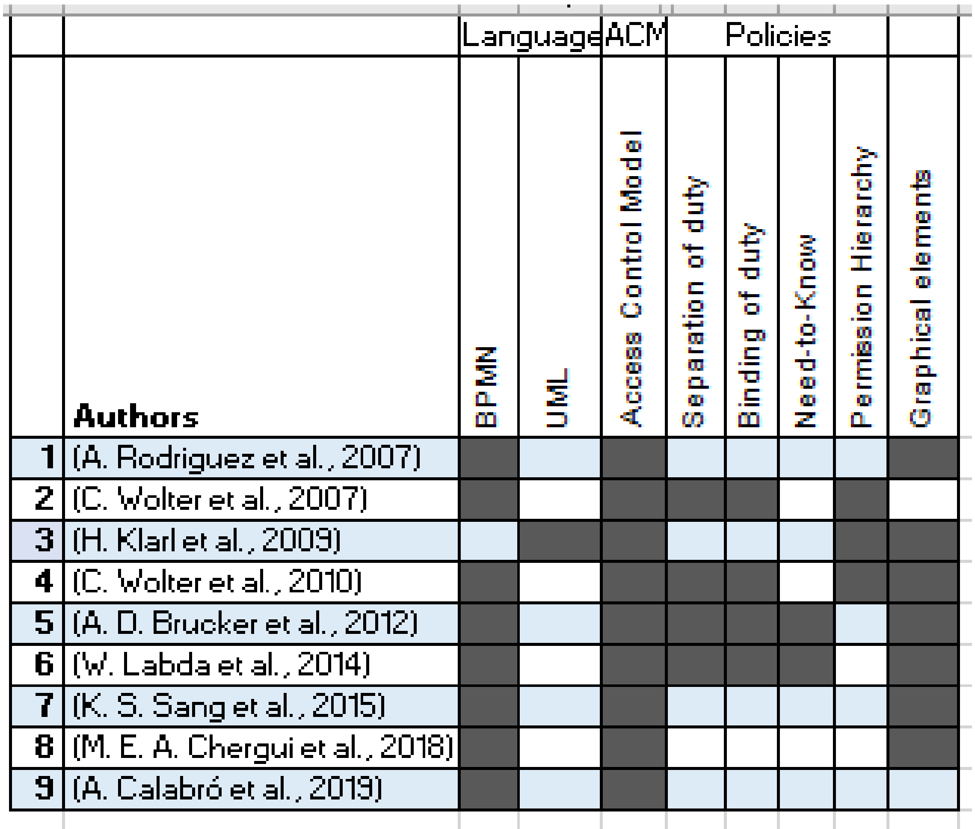
In [27], the authors propose a solution for ”exploit the BPMN models for assisting in the design, development, maintenance, and verification of a system to comply with the General Data Protection Regulation (GDPR) requirements” [27]. Because current access control mechanisms do not satisfy GDPR requirements, the authors propose a methodology for combining, merging, and integrating ”the access control systems into the business processes to address different aspects of the GDPR compliance problem” [27]. The methodology is composed of several steps and uses the standard Extensible Access Control Markup Language (XACML) to his advantage, among them are: ”Gather authorization requirements”, ”Identify required attributes”, ”Author the authorization policies”, ”Test the policies”, ”Deploy the architecture” and ”Deploy policies”. The first step of the methodology is to gather all the requirements and express them in terms of natural language. The second step will identify the activities that are affected by the GDPR requirements, and will be also in charge of substituting them with sub-processes to make it regulatory compliant with the GDPR. The third step is in charge of transforming the natural language requirements statements into XACML policies. The fourth step, will test this policies to see if they meet the GDPR requirements. Lastly, the fifth and the sixth step are in charge of deploying the XACML-based access control system and the policies to rule the access.

In [33], the authors proposed a solution for integrating security requirements, such as authentication, access control, authorization and non repudiation, among others, into BPMN for healthcare business processes. To do so, the authors' solution consists on adding new types of boundary and intermediate events and activities to represent these requirements. The access control used in the paper are boundary events and can be applied to activities or groups to limit the access to already authenticated users. Together with these requirements, the authors introduced a new flow object to BPMN, that receives the name of security indicators and is in charge of indicating the level of security strength of the process and its strength level relies on the security event in the process [33].

Lastly, in [34], the authors propose another solution for representing security requirements in BPMN. they provide ”a valid BPMN extension with complete set of security concepts derived from cyber security ontology to enable the modelling of the security requirements”, among which are access controls. The paper presents a good related work, that was used to confirm that some of the selected papers where relevant in the topics of integration of access controls to business process standards.

From the papers, it can be extracted, that there is a separation between business processes and security requirements, such as access controls. Mainly because the modeling languages, such as BPMN, were created to represent the flow, and the coordination of the business process when modeled. Thus, they do not have a representation of these types of requirements. Figure 3.5 shows an overview of the features of these papers. The grey cells indicate the languages, access control model, policies, and other properties that each paper contains and solves (e.g., for A.Rodriguez et al., 2007 the paper uses BPMN, contains access control models in the solution, and uses graphical elements). The first two columns indicate the chosen language for implementing the solution. The following column indicates, if it is being use any access control mechanism in the paper. The next three columns indicate if the paper solves these security requirements with the solution presented. Lastly, the column *Graphical elements* points out if visual elements were added to the expansion of the language used in the solution.

As it can be seen by the table the integration of security requirements, such as access control models, is an area of research relevant, since papers about the topic have been written and published from 2007 to 2019. Regarding access control models, the predominance of the papers only implements models



		Language		ACM	Policies				
	Authors	BPMN	UML	Access Control Model	Separation of duty	Binding of duty	Need-to-Know	Permission Hierarchy	Graphical elements
1	(A. Rodriguez et al., 2007)	■		■					■
2	(C. Wolter et al., 2007)	■		■	■	■		■	
3	(H. Klarl et al., 2009)		■	■				■	■
4	(C. Wolter et al., 2010)	■		■	■	■		■	■
5	(A. D. Brucker et al., 2012)	■		■	■	■	■		■
6	(W. Labda et al., 2014)	■		■	■	■	■		■
7	(K. S. Sang et al., 2015)	■		■					■
8	(M. E. A. Chergui et al., 2018)	■		■					■
9	(A. Calabró et al., 2019)	■		■					

Figure 3.5: Overview of the related work

based on roles, which limits the level of expressiveness in terms of organizational policies. The papers that also implement models based on attributes (i.e., they use ABAC) and explicitly say it are, [27] that applies ABAC in the specific context of GDPR compliance, using the advantage of XACML to define ABAC policies. Lastly, [31], uses graphical elements for the representation of policies based on roles and attributes, being later applied to activities. However, the ABAC policies have a focus on representing the requirements of separation and binding of duty and because they are directly applied in activities and data objects this forces the definition of an ABAC policy in each activity, which withdraws the activity categorization that lanes do. Concerning graphical elements, most of the papers use extended visual representations for representing the needed requirements, however, this can obstructs the integration of the approaches in already existing business process modeling tools [34].

In conclusion, there are different proposed solutions for integrating security requirements into business processes, and each of them does it with an objective in mind, that go from a need to represent security requirements in BPMN; a need to represent identity management requirements in business process for service-oriented architectures; and also a need for a solution for compliance of the GDPR. Thus, the topic has a variety of areas of use and is still a relevant, as can be seen by the years of publications of papers in figure 3.5. In the developed solution of this work, BPMN is used as the language to expand

and integrate access control models and to implement topics introduced by the papers, such as role hierarchies or the concept of the ’need-to-know’. The models chosen for the solution are RBAC, and ABAC, which contribute by giving more simplicity or more flexibility for representing organizational policies, respectively. Lastly, a tool was developed to check the business processes modeled regarding a set of rules defined by the expansion of the language; and give a query system that will provide actors a checklist that combine with the business process diagram will prevent non-complying behaviors related to the resource’s policies of the organizations.

4

Solution

Contents

In this chapter, a theoretical solution for the framework of the BPMN language will be presented. Along with the implementation of the framework, in Atlas, and the developed tool. The chapter is divided into three sections, each of them dedicated to one of the previous topics.

4.1 Theoretical Approach: BPMN Meta-model

Business processes are defined as a collection of inter-related events, activities, and decision points that involve several actors [2]. BPMN provides a graphical representation of the coordination and flow of business processes easy to understand by modelers and other viewers [4]. When extending the language it is important not to change the footprint of any existing flow element, such as events, activities, and gateways [4]. In other works, such as the ones presented in the previous chapter, some of the solutions consisted of adding new elements to represent security requirements, for instance, by adding symbols or using other visual elements to indicate the roles of RBAC models. The solution created in this work does not add new visual elements to represent access control models. However, new information will be added to BPMN elements that already existed. This was done because Atlas does not allow the creation of new visual elements and also with the objective of not interfering with the graphical representation of the language.

Before making the extension of BPMN to admit access control models an interpretation, and simplification of the original BPMN meta-model (see Chapter 2, figure 2.1) was done. In this interpretation, new relations between elements were expressed (e.g., between lanes and activities), and some elements were directly substituted by an association relation (e.g., Data Association). This was done with the objective of simplifying the meta-model for better integration of the access control models. This is presented in figure 4.1.

The meta-model with the integration of the access control models was done based on the interpretation. Figure 4.2, represents the interpretation of the BPMN meta-model with the extension to accept access control models. The green-colored entities in the diagram are the new elements added to the BPMN model. These correspond to the classes related to the access controls and do not have a visual representation when modeling. The idea behind it was to use the concept of Lane and enlarge his meaning. Lanes are partitions used to organize and categorize activities, normally representing internal roles or departments [8]. When lanes are defined in BPMN, they represent participants within the business process and ”each of these participants is shown as a separate lane containing the activities performed by the participant in question” [2]. If the meaning of lanes is enlarge with the information provided by access control models, we will be able to define the what was assigned to each of the participants as well as the required permissions to access. Hence, in this thesis access control models are consider a type of lane where a permission is defined to access to the activities of the lane.

Concerning these permissions, notice that there are two types. This is because RBAC only needs to know about the roles of the actors that can allocate the tasks; and ABAC not only can accept roles, but also other information about the subject, the object, or the environment, allowing a finer level of granularity and create policies that are combinations of attributes [35]. The attributes considered, in this case, were based on the National Institute of Standards and Technology's (NIST)[1]. These are:

- **Subject**:
    - **Certification:** Indicates official documents that the person has achieve.
    - **Division:** Indicates the departments or groups where the person belongs to.
    - **Email:** Indicates the different emails of a person.
    - **Training:** Indicates the skills that a person has.
    - **Role:** Indicates the different positions that a person has in a organization.
- **Object**:

[1] https://www.youtube.com/watch?v=cgTa7YnGfHA&feature=emb_title , Accessed = 03/11/2020

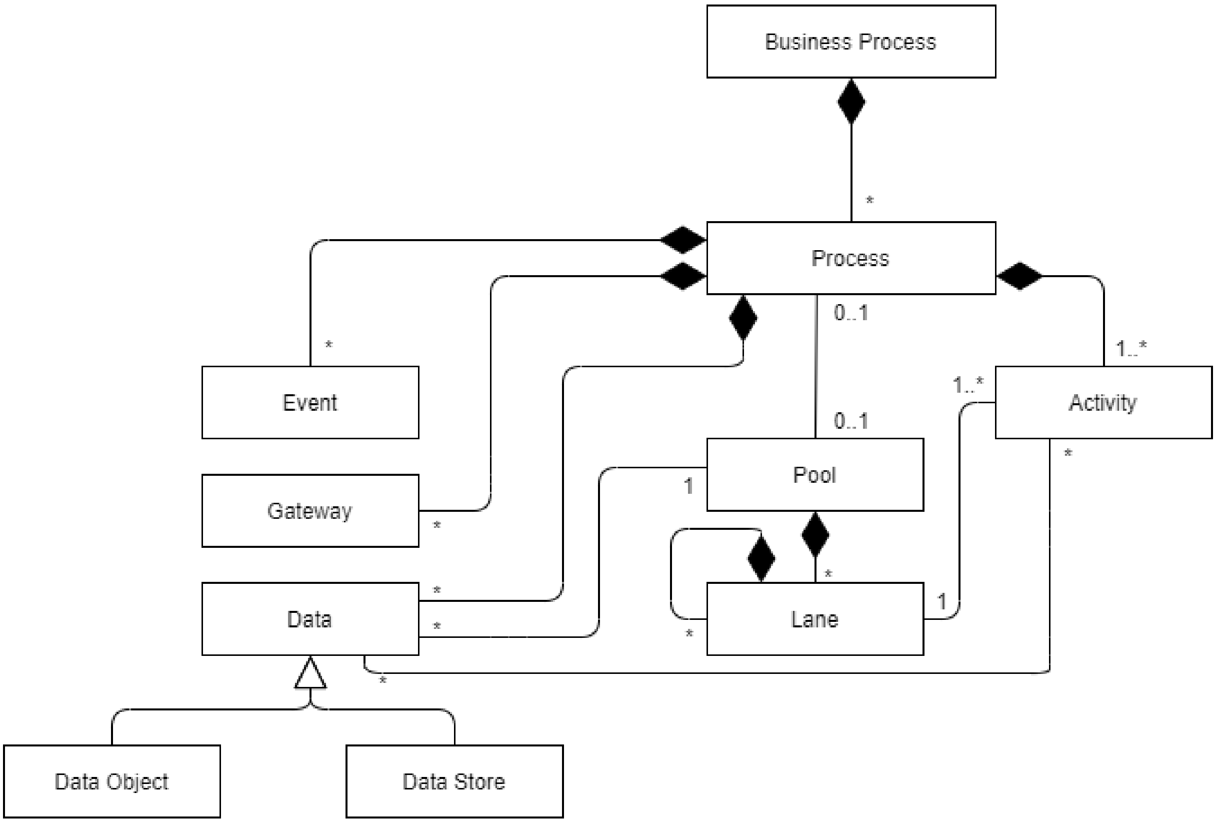


Figure 4.1: BPMN meta-model's simplification

  - **Project:** Indicates the projects that a person is involve in.

- **Environment**:

  - **Location:** Indicates the places where a person works.

Both of these types of permissions have associated business functions, which represent lower-level activities in which there is a privilege. For example, the business functions of the activity ”Approve Travel Proposal” of a Request Travel Approval business process, would be: ”Approve Budget” and ”Approve Duration”; The type of model chosen to execute these business functions depends on what is the intended policy to use in the process. If the business functions should be done by a person with a Manager role, the RBAC model would be suitable. However, if the policy is more elaborate, such as, for instance, ”A manager that belongs to the Travel Approval division of the Lisbon’s franchise” the most suitable model would be ABAC, since it will allow the specification of other factors, such as the location.

Notice that is possible to have access control models that aggregate other access control models. This will happen if the business process has nested lanes, i.e., parent lanes with sub-lanes inside. Nested lanes are used if it is needed to implement a hierarchy in terms of the permissions of the models. For instance, in figure 4.3, the ”Manager” lane will inherit all the business functions of the activities that the permission of the sub-lanes ”Worker 1” and ”Worker 2” have.

Regarding data objects and data stores, ”they represent information flowing in and out of activities” [2], and ”provide a mechanism for activities to retrieve or update stored information that will persist” beyond the Process, respectively [8]. Atlas uses bpmn.io for modeling BPMN. During the time, data objects and data stores suffered changes regarding how they are modeled in bpmn.io. In 2016, in bpmn.io, both of these data needed to be contained by a pool. Later, in 2018, this changed, and in the current version, only data objects are required to be modeled inside pools[2]. This was not taken into account when developing the solution. Thereupon, in the meta-model is an association between both of these data elements and pools.

Lastly, because events correspond to things that happen atomically, that is, they have no duration [2], and gateways have the objective of controlling the flow of the process through sequence flows [8]. They were not considered to have anything relevant to the access control models and solution, other than for modeling the processes.

[2] https://bpmn.io/blog/posts/2018-bpmn-js-2-0-0.html Accessed= 09/2020

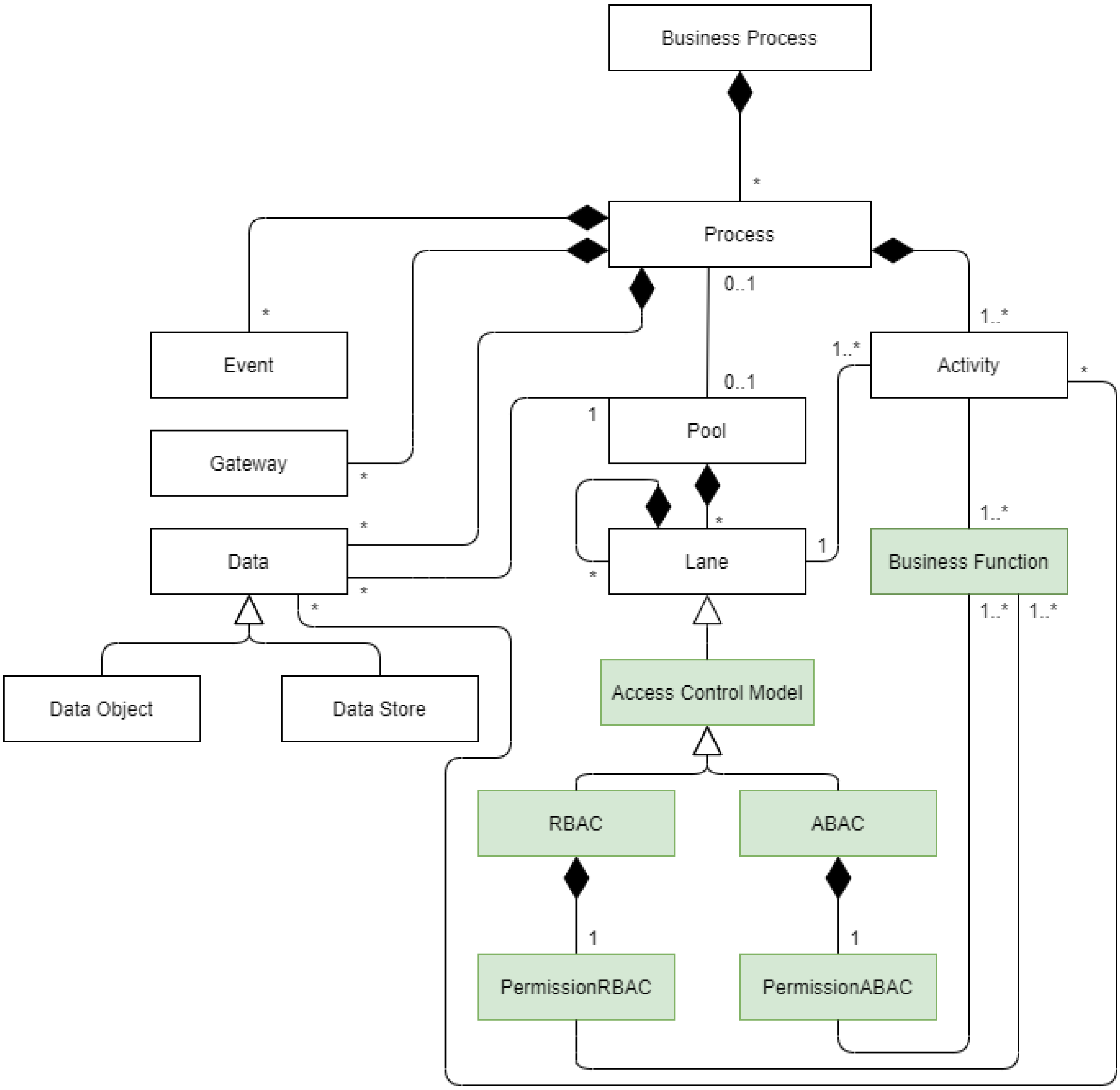


Figure 4.2: BPMN meta-model's extension with access control models

4.2 Atlas: Metal-model Implementation

The developed solution consists of implementing the meta-model presented using the Atlas project and developing a tool that counts with a part for validating processes that use this new extension of BPMN and with a query system to answer questions about the processes.

Atlas is a project that aims to endow organizations with a bigger capacity for managing and planning their IT. It is a solution for simplifying the process of creating and sustaining the IT architecture of the organizations. The functionalities of this tool used for this work are related to the modeling of business processes using BPMN and the Rest API that Atlas has. As explained in the previous section, Atlas is

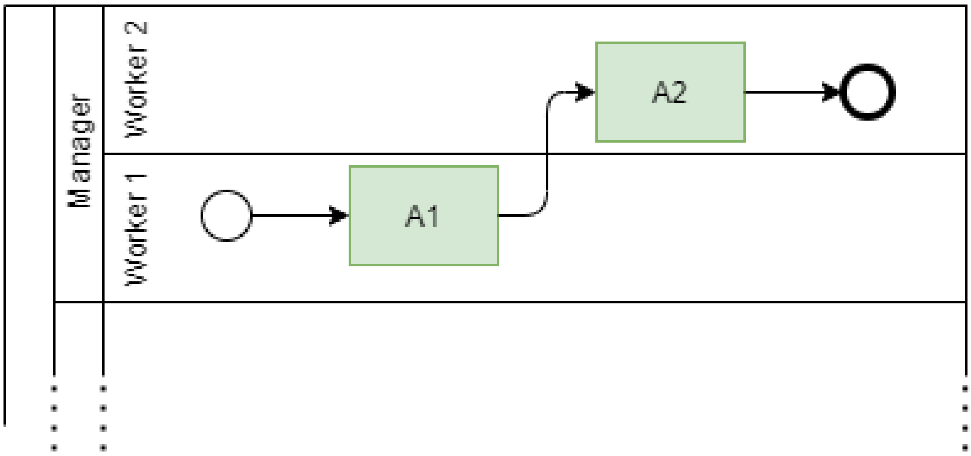


Figure 4.3: Permission inherit example

powered by bpmn.io (bpmn.io/), a tool that allows the visualization and easy creation and edit of BPMN diagrams using BPMN 2.0.

Atlas functions all based on classes. The elements of the business process for Atlas are just instances of classes that are mapped to the BPMN elements. However, it is possible to model a business process without classes mapped, but for Atlas to give meaning to the business process elements and retrieve the information about the process when asked through his API, they need to be mapped to classes. Therefore, for this work, two steps need to be done before modeling business processes. These are: create classes and objects, and map them to BPMN elements. Figure 5.7 shows the Atlas's interface for creating classes and objects.

To get the created business process, therefore the objects, outside Atlas, the Rest API has to be used. All the processes modeled in Atlas belong to a repository. To each request done to the API, it's needed to indicate the repository where the wanted information is. To which the API will respond with a JSON with the information of the process asked. However, this information does not come all together at once. Equally to business processes, there are layers, and for instance, if using the proposed meta-model as an example, if the information about a pool is requested, the JSON retrieved will only have the names and IDs of the lanes in the pool. For instance, if the request was to get information about the lane *Finance* from figure 5.10, the retrieved JSON, among other things, would be as presented in Appendix A. As it can be observed, the information regarding activities, data objects, data stores, or business functions is not presented, as well as information regarding the permission of the lane. To get it, a new request using the IDs of these elements is needed.

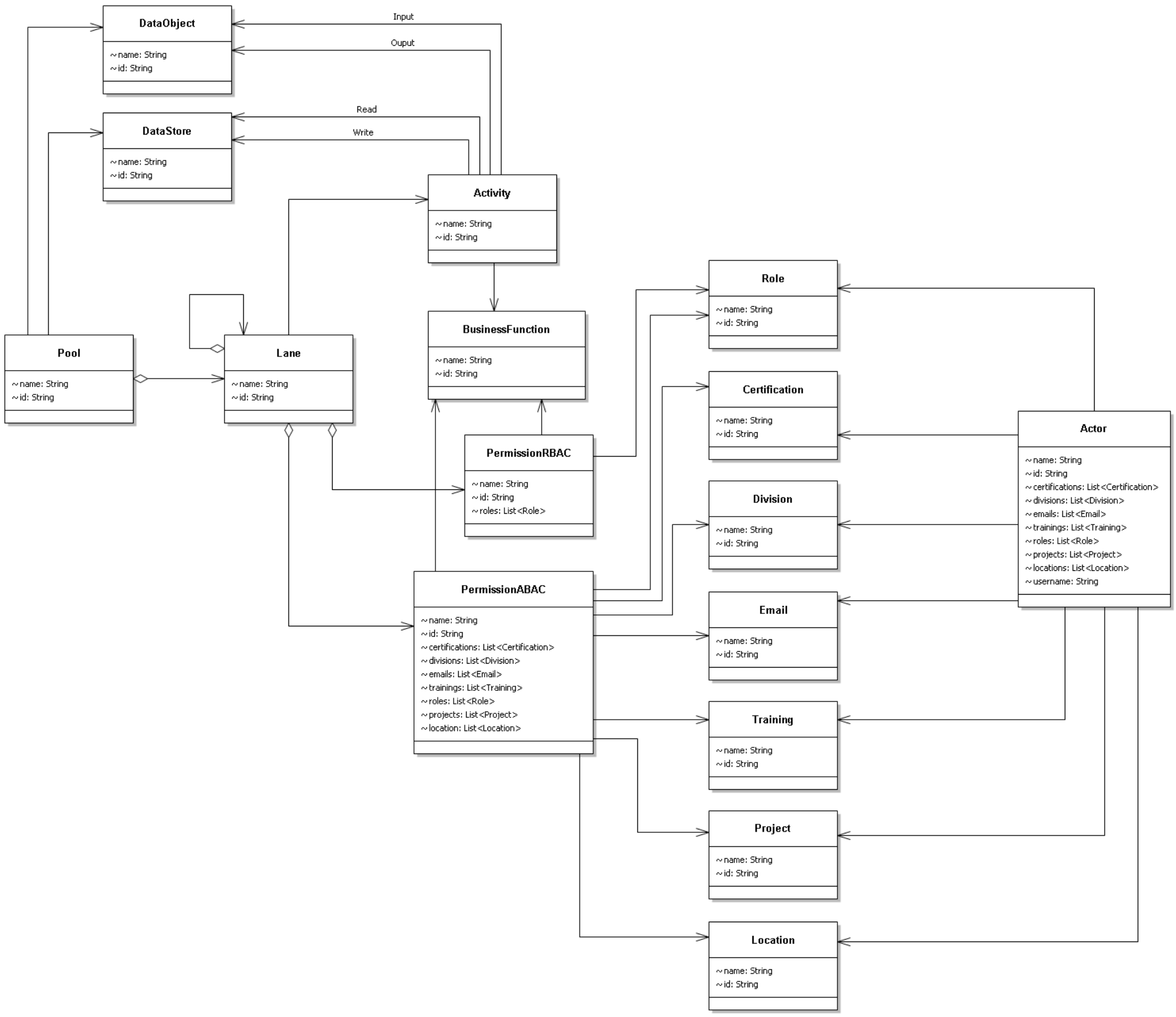
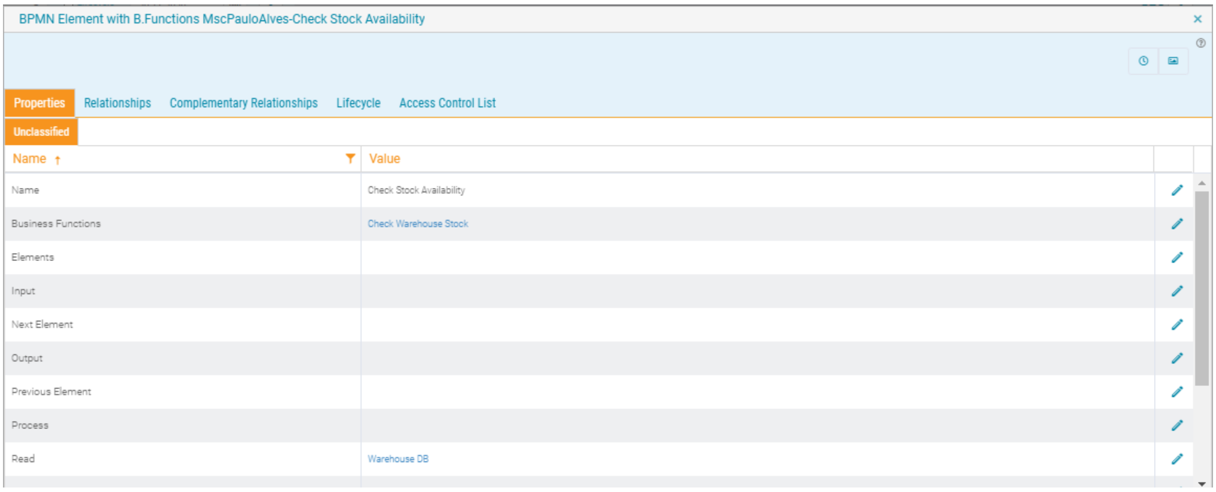


Figure 4.4: Class relation diagram in Atlas

4.2.1 Implementation

In this sub-section, the implementation of the theoretical meta-model, in Atlas, will be explained. Before beginning, different approaches were done to better represent the model from figure 4.2, due to lack of knowledge of Atlas, and because Atlas does not support inheritance between classes. Therefore, in terms of implementation, the meta-model is a variation of what is represented in figure 4.2. This sub-section presents the final solution for implementing the theoretical solution in Atlas. Figure 4.4 represents the class relation diagram created as is, in Atlas. To explain how the model is in Atlas, the Order Fulfillment business process, shown in figure 1.1, will be used as an example.



BPMN Element with B.Functions MscPauloAlves-Check Stock Availability

Name	Value
Name	Check Stock Availability
Business Functions	Check Warehouse Stock
Elements	
Input	
Next Element	
Output	
Previous Element	
Process	
Read	Warehouse DB

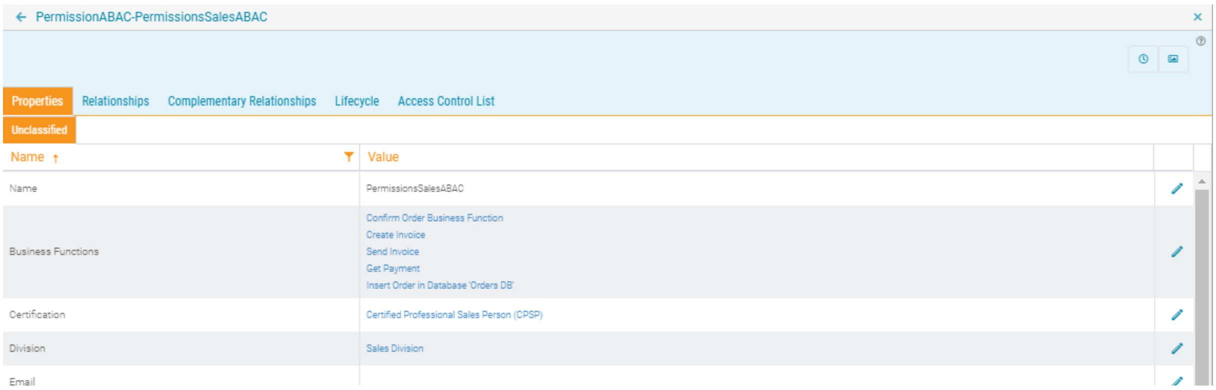
Figure 4.5: Check Stock Availability activity

4.2.1.A Activity Class

Starting with activities, the Atlas class receives the name of BPMNElement, and these are associated with business functions, data objects, and data stores. Atlas does not support data inputs and data outputs. Thus, what indicates, if a data object is an input or an output, is the data association that links the activity to the data object. This is the reason why there are two different associations between data objects and activities, one dedicated to data objects that are inputs, and another dedicated to data objects that output. Data association also gives the same meaning when there is a connection between data stores and activities. However, instead of using input and output to represent the relations, the names read and write are used. Figure 4.5 shows the activity *Check Stock Availability* of the Order Fulfillment business process. The activity has the business function *Check Warehouse Stock* and uses the *Warehouse DB* for reading information about it.

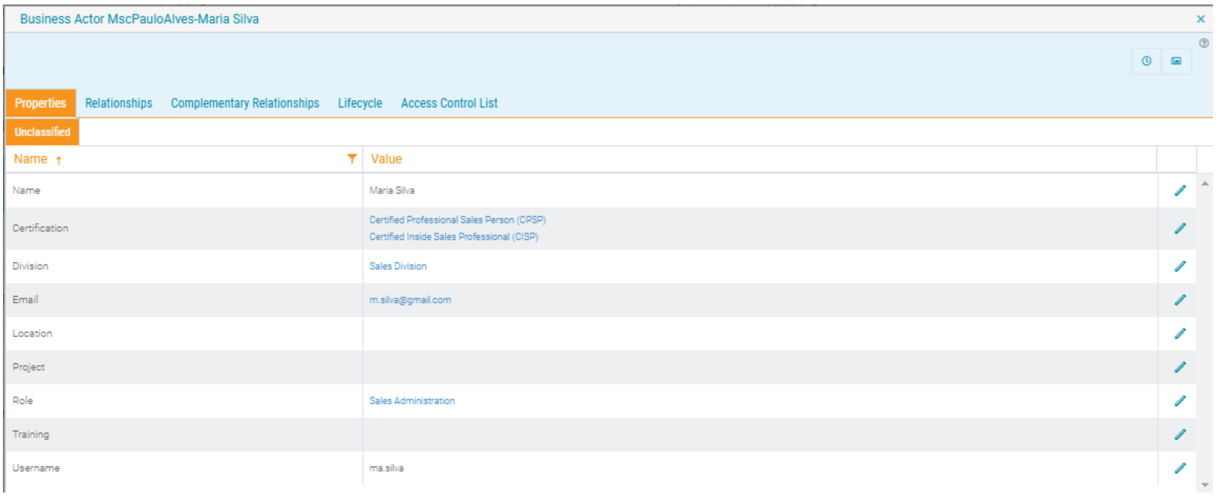
4.2.1.B PermissionRBAC, PermissionABAC, and Actor Classes

Continuing with the two classes dedicated to express permissions (i.e., PermissionRBAC and PermissionABAC), these classes serve to join activities' business functions and actor's attributes. The main goal of the class is, indicate the attributes that a given actor needs, to perform a particular set of business functions. What was intended was to limit one permission per lane, to represent the idea that a lane symbolizes one set of activities assigned to an actor. However, there wasn't an option to do this in Atlas, therefore, this was taken care of in the developed tool. By choosing one type of permission to a lane it is being defined, the access control model that a given lane is. Using the Order Fulfillment as an example, figure 4.6 shows the object *PermissionSalesABAC*, which is of type PermissionABAC, and implements the policy:



Name	Value
Name	PermissionsSalesABAC
Business Functions	Confirm Order Business Function Create Invoice Send Invoice Get Payment Insert Order in Database 'Orders DB'
Certification	Certified Professional Sales Person (CPSP)
Division	Sales Division
Email	

Figure 4.6: Sales Permissions for Order Fulfillment business process


Business Actor MscPauloAlves-Maria Silva
Properties
Relationships
Complementary Relationships
Lifecycle
Access Control List
Unclassified

Name	Value
Name	Maria Silva
Certification	Certified Professional Sales Person (CPSP) Certified Inside Sales Professional (CISP)
Division	Sales Division
Email	m.silva@gmail.com
Location	
Project	
Role	Sales Administration
Training	
Username	ma.silva

Figure 4.7: Example of object of the Actor class

- *Only actors that have the Certified Professional Sales Person (CPSP), and belong to the Sales division of the organization, can confirm orders, emit invoices, receive client's payments, and archive orders.*

Therefore, the object contains the attributes *Certified Professional Sales Person (CPSP)*, and *Sales Division* and states that the actors with this attributes are able to perform the business functions *Confirm Order Business Function*, *Create Invoice*, *Send Invoice*, *Get Payment*, and *Insert Order in Database 'Orders DB'* that belong to activities that are present in the lane *Sales*.

Following the idea that permissions join actors' attributes and business functions, the other important class to explain is the class Actor. This class aims to express the properties of the actors present in an organization. Figure 4.7 shows an example of an actor with adequate properties to perform the business functions present in the permission *PermissionSalesABAC*.

4.2.1.C Lane Class

Regarding the Lane class, it is composed by the attributes: BPMN Elements (i.e., Activities), Permissions and Sub-lanes.

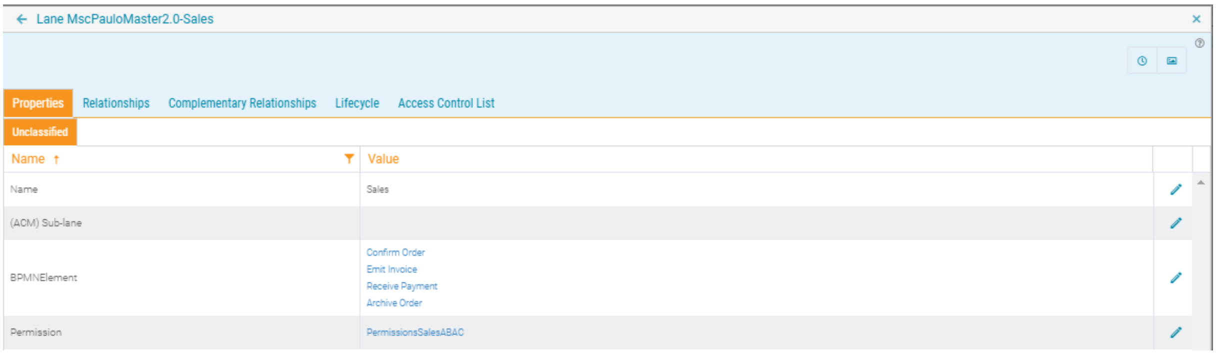
The Sub-lanes attribute indicates the lanes inside the lane. In this work, every time a lane has sub-lanes it is considered a parent-lane. Parent-lanes work differently than sub-lanes. When a lane is parent-lane, it must not be associated with activities, therefore the attribute Activities in the Lane class should be left empty. On the other hand, if a lane is the last of the sub-lanes, then, the attribute should have all the activities present in that sub-lane. For instance, in figure 4.3, *Manager* is a parent-lane, and *Worker 1* and *Worker 2* are the last sub-lanes. Thus, *Manager* must have the attribute that corresponds to the activities empty; *Worker 1* should have the activity *A1* in the attribute; and *Worker 2* should have the *A2*. The reason for this constraint is that it was not found a way to create two different classes for representing parent-lanes and sub-lanes in Atlas. Therefore, the same class Lane was used to represent all types of lanes.

The attribute Permission designates the permissions of the lanes and is mutually exclusive. It can be of type PermissionRBAC or PermissionABAC. Depending on which of these two classes is used for it, the lane will be of type RBAC or ABAC. Although it is possible to define multiple objects for the attribute at the same, it will be required for lanes to only have one object associated with the attribute for the developed tool to work. This is due to the idea that one lane represents one set of activities assigned to an actor. Parent-lanes also have different behavior in terms of permissions, since they are not required to have one if desired. If a parent-lane does not have a permission it causes the permission hierarchy property to not be implemented. In contrast, the last sub-lanes should always have a permission. If not, business functions will not be executed, since they belong to nobody.

Figure 4.8, shows an example of an object lane. It represents the Sales lane of the Order Fulfillment business process. The object does not have any sub-lanes, is composed by the activities: *Confirm Order*, *Emit Invoice*, *Receive Payment*, and *Archive Order*, and has the permission *PermissionSalesABAC* from figure 4.6.

4.2.1.D Pool Class

To conclude, as in the theoretical approach pools are constituted by lanes, and are associated with data objects and data stores. This association received the name of ”owns” for later parts of the document and is represented in the model due to the reasons stated in the section 4.1. Figure 4.9, shows the object Seller’s Organization pool. This object has two lanes (Sales and Warehouse & Distribution) and owns two different databases (Warehouse DB and Supplier Catalog).



Name ↑	Value
Name	Sales
(ACM) Sub-lane	
BPMNElement	Confirm Order Emit Invoice Receive Payment Archive Order
Permission	PermissionsSalesABAC

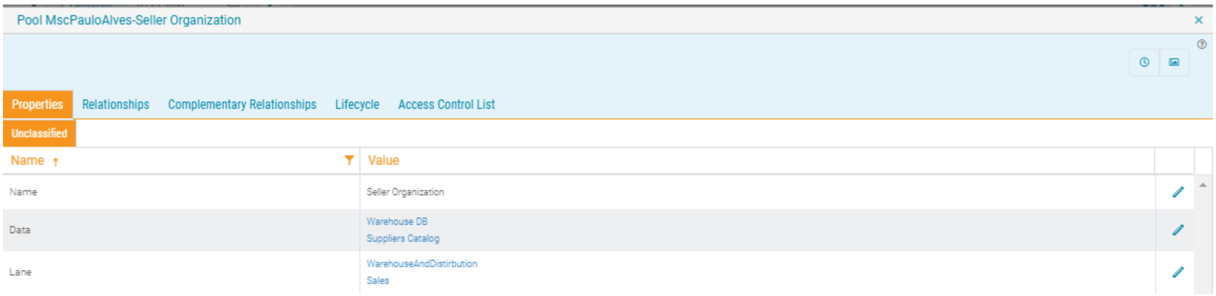
Figure 4.8: Example of object of the Lane class

4.2.2 Lessons Learned

Due to the little knowledge of Atlas, two different approaches were developed before the final one. In each of them, it was tried to better represent the theoretical class diagram shown in figure 4.3, correct the limitations that previous approaches had, and add more content to the solution. Figure 4.10 shows these two approaches developed before the final solution.

The first approach (figure 4.10 a)) overall was very simple and was a start-point for using Atlas. Thus, it had many limitations. These were: RBAC was the only model implemented, which limited the flexibility and the number of ways to express policies; the approach only considered parent lanes to give roles. Therefore, the roles were only defined in the top lanes, causing the hierarchy between permissions not possible. Lastly, another limitation was that the actors had to be defined in the lanes of the processes. Which was not a good idea since every time a new process for an organization, was modeled, the actors had to be included in it and took away the objective of integrating the access control models into business processes since actors simply could be inserted in the lanes where they had activities to execute.

The second approach (figure 4.10 b)) corrected the limitations of the previous one. In it, ABAC was included which gave more expressiveness for modeling policies; nested lanes were also included, which allowed the implementation of permission hierarchies; actors started to be modeled as presented in the


Pool MscPauloAlves-Seller Organization
Properties
Relationships
Complementary Relationships
Lifecycle
Access Control List
Unclassified

Name ↑	Value
Name	Seller Organization
Data	Warehouse DB Suppliers Catalog
Lane	WarehouseAndDistirbution Sales

Figure 4.9: Example of object of the Pool class

final solution. Therefore, they no longer needed to be included in the business process when it was being modeled, and the concept of a business function was introduced with the objective of creating a relationship between access control models and activities.

In this solution, lanes were a middle step to define what was the access control model that a lane would follow. Lanes that were RBAC models were associated with activities, had sub-lanes, and permissions which were associated with activities' business functions, and with the roles that could execute them. Similarly, ABAC lanes were associated with activities, had sub-lanes, and permissions which were associated with activities' business functions, and with the attributes described in section 4.1. The combination of these attributes defined who could perform the business functions. However, if wanted, permissions could not be defined in parent-lanes (for both RBAC and ABAC lanes), and in those cases, the permission hierarchy was not considered. The main idea was to give more flexibility when modeling processes. However, this was not clear and could confused, since it was possible to have lanes that were RBAC or ABAC but without permissions. This was the flaw in the approach. Lanes were just objects of a class that had the purpose of choosing the desired access control model. Therefore, the class did not respect the definition given by BPMN (i.e., it was not meant to categorize activities). This flaw was caused by the ignorance of the possibilities that Atlas offered, and was corrected in the last version. This can be observed in figures 4.11, and 4.12, that show the lane Warehouse & Distribution object, and RBAC model that the lane followed in the Order Fulfillment business process (figure 1.1). In figure 4.11 the object of the lane Warehouse & Distribution is shown, and it is being chosen for the lane to follow a RBAC model. Then, in figure 4.12, is where the properties of this RBAC model are made explicit. Like the final solution, the field *BPMN Element* indicates the activities that are in the lane. The field is empty because the lane is a parent-lane, and putting activities here will cause a fault during the validation of the process in the tool developed. The field *Permissions* indicates the permission objects associated with the lane. Lastly, the *Sub-lanes* field specifies the sub-lanes that the lane Warehouse & Distribution. By clicking on *Warehouse Worker*, the same interface of figure 4.11 will be shown but with the information regarding the *Warehouse Worker* object.

4.3 Tool: Mapper, BusinessProcessChecker and Queries

The other part of this work consisted of the development of a tool using Java 1.8 for checking the business process and create a system based on queries that answered questions about the business processes that the actors of the process have, to help keep a correct behavior of the process.

The extension of the BPMN language caused the integration of new information to the already existing elements of the language. Therefore, it needs to be treated and given some more meaning. It was with this purpose that the tool was also developed. Figure 4.13 shows the relation between the classes of

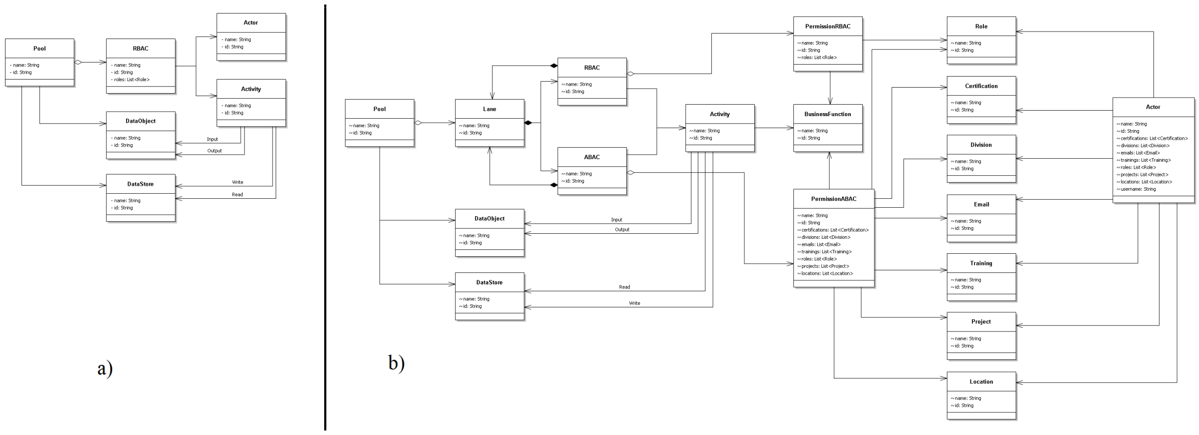
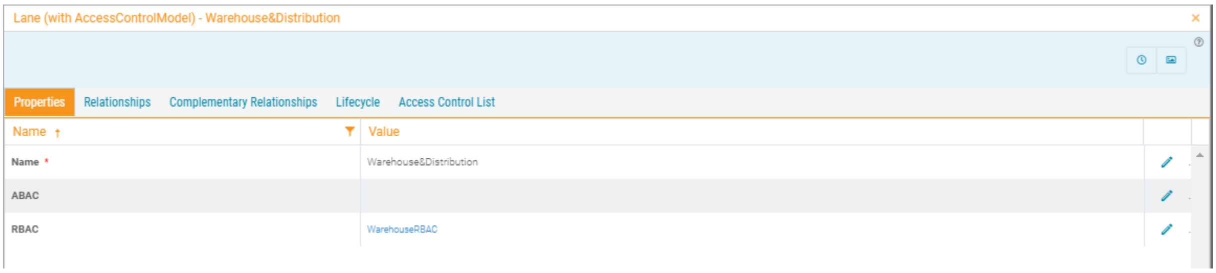
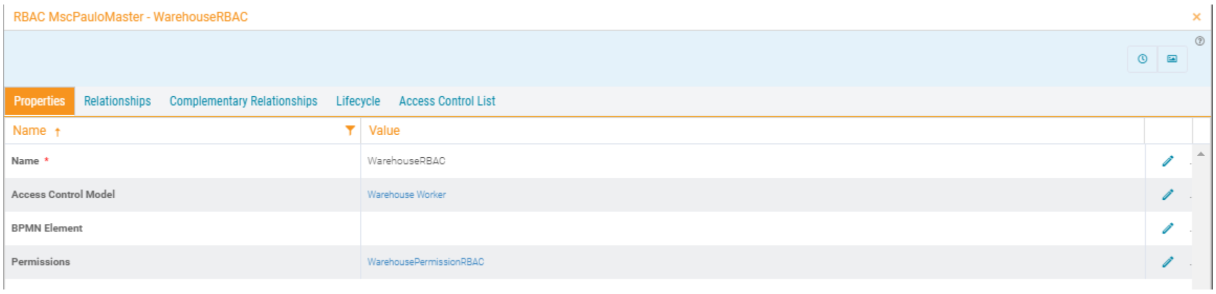


Figure 4.10: Class relation diagram in Atlas for the first (a) and second (b) approach


Lane (with AccessControlModel) - Warehouse&Distribution

Name ↑	Value
Name *	Warehouse&Distribution
ABAC	
RBAC	WarehouseRBAC

Figure 4.11: Object Warehouse & Distribution of type Lane


RBAC MscPauloMaster - WarehouseRBAC

Name ↑	Value
Name *	WarehouseRBAC
Access Control Model	Warehouse Worker
BPMN Element	
Permissions	WarehousePermissionRBAC

Figure 4.12: Object WarehouseRBAC of type RBAC

the created tool.

As it can be observed from the figure, there is a great part of the classes of the diagram equal to Atlas's class diagram from figure 4.4. The reason behind it is that these classes are needed to map the information retrieved by the Atlas's API. The colored classes, on the other hand, are totally new and each one has a different objective and functionality. These are:

- **Manager:** This class has the goal of interacting with the other important modules (i.e., it manages all the operations). All the commands to execute are sent to the respective class through here, in order to be executed. This class is where the identity of the user, that wants to login into the tool,

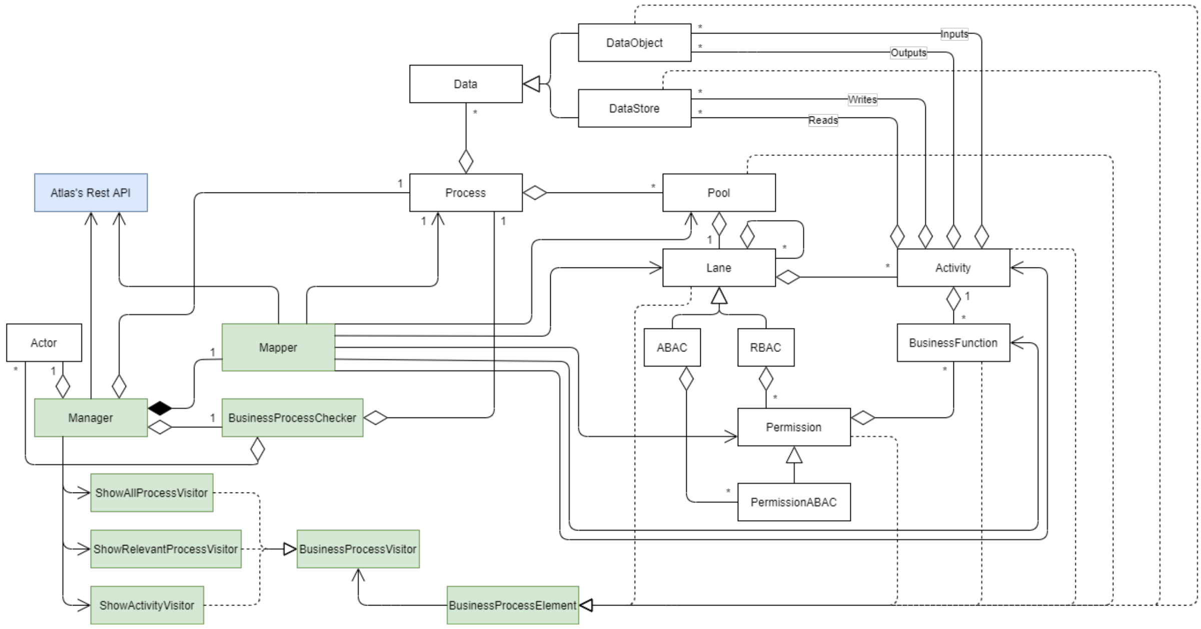


Figure 4.13: Tool's class diagram

will be validated. To do that the Manager class uses Atlas's API to his advantage. The class sends a login request to the API URL[3] and contains a JSON file where the parameters ”Username” and ”Password” of the user are given. These parameters are the ones given by the user at the login of the tool. This operation retrieves a JSON file with a response code (e.g., HTTP 401 Unauthorized), and if the response code is valid (i.e., HTTP 200 OK), an authentication token.

Once the login is successful, the user has available a form for writing the name of Atlas repository where the business process is, and the name of the business process that he wants to use the tool with. Immediately after these fields are indicated, the class performs two operations. The first one consists of getting the repository ID that Atlas has given to the repository. This is done by using the URL[4] to which Atlas returns a JSON File with the ID, among other things. Then, the class carries out a similar operation for loading the process, using the URL[5]. Once the business process ID is obtained, the Manager class calls the Mapper's class method, *loadProcess*, that is in charge of creating objects for each element of the business process with the information retrieved by the API.

After the business process is mapped in the tool, the Manager class, calls the BusinessProcess-Checker, which validates if any mistakes in terms of syntax or semantics of the BPMN framework

[3] https://atlas.linkconsulting.com/rest/login

[4] https://atlas.linkconsulting.com/rest/repository/list/?filter=[where][name][eq][<repositoryName>]

[5] https://atlas.linkconsulting.com/rest/object/list/?filter=[where][objectClass.repository.id][eq][<repositoryId>], [where][objectClass.name][eq][Business%20Process],[where][name][eq][<businessProcessName>]

were made when modeling. These mistakes are translated to faults or warnings depending if they allow the process to run correctly or not. Once the process is valid and there are no faults, the user can use the queries and choose between the available ones. When a query is selected, and the relevant information for the query is given, the Manager class calls one of the classes dedicated to the queries (i.e., ShowAllProcessVisitor, ShowRelevantProcessVisitor, and ShowActivityVisitor).

- **Mapper:** This class has the purpose of receiving the information about a process modeled in Atlas and creating the respective objects in Java. The execution of this class starts when the Manager class calls the Mapper's method *loadProcess* with the business process ID obtained from Atlas in the previous step. Then, the class uses the URL[6], alternating the ID to obtained all the elements in the different layers of the process. The first ID (i.e., business process ID) gives us the IDs of the pools of the process; using every pool ID, the lanes and the data objects and data stores of the pool will be known as well as their IDs. Then, the lanes' IDs will be used to obtain the activities, permission, and sub-lanes to which the lane is associated. This procedure will be done for the sub-lanes and all the other elements of the process mapped to a class in Atlas until the Atlas's business process is totally mapped to the tool's classes. In the case of the lanes a recursion function was implemented, since every lane object has the possibility of having another lane associated. Finally, becuase the permission entity associated to the lanes defines the type of lane that a lane is (i.e., RBAC, ABAC or normal lane), the mapper when encountering a permission (or not) creates one of those types. Thus, for instance, for the *PermissionSalesABAC* the Mapper will create a lane of type ABAC.

  Once every object of Atlas exists on the tool, the method, *loadProcess*, returns, and the Manager proceeds to call the BusinessProcessChecker class as described.

- **BusinessProcessChecker:** Because there was an extension in the language, it is required to give meaning to everything new of the language. This resulted in the creation of rules. This class has the objective of verifying if these rules are respected in the business processes models that come from Atlas. These rules, if not satisfied, produce faults and warnings depending on the severity of the mistake. These rules are:

  - **Rules that produce faults:**
    - A business function always must have at least one actor that can execute them.
    - There must only be one permission per lane.
    - If the lane is not parent-lane then it must have a permission.
    - Every business function must be associated to a permission.

[6] https://atlas.linkconsulting.com/rest/object/properties/list?filter=[where][object.id][eq][<Id>]

    - The business function must be associated to a permission of the lane where his activity is modeled.
    - Every data object and data store should be associated to the pool where they are modeled.
    - An object lane that is parent-lane, must not have activities associated.
  - **Rules that produce Warning:**
    - Data objects or data stores associated to a pool object should be used in the process.

  In case they are satisfied, the class returns, and the queries of the tool become available to the user of the tool. Otherwise, the class returns the list with all the mistakes made while modeling and indicating the business process elements where they can be corrected. Figure 5.11 shows an example of faults and warnings of a process.

- **Queries:** To implement the queries, a visitor pattern design, was used. The pattern allows easing the creation of queries when needed since it is only required to create a new class for a new query, and does not force to modify any part of the already created classes. The classes ShowAllProcessVisitor, ShowRelevantProcessVisitor, and ShowActivityVisitor represent each of the queries implemented.

  The first class when called shows the entire business process and the components associated to each class. It starts with the pools and descending the levels of the process (i.e., lanes, sub-lanes, and activities). To obtain the output of given by this class, the user of the tool has to select the option ”Show Complete business process” when running the tool.

  The second class implements the concept of *need to know* explain in chapter 2. The class is in charge of crossing the information of the actor of the business process with the business process, to know the business functions that the actor can access and are assigned to them to execute. Before calling this class, the Manager class requests Atlas to retrieve the information about the user logged in. Once Atlas retrieves it, the class ShowRelevantProcessVisitor is called with the information and crosses it with the information of the permissions of the business processes loaded in the tool. The output is a list with all the activities and business functions that are assigned to the actor to execute, organized by pool and lane. To obtain the output of given by this class, the user of the tool has to select the option ”Show Business Process for Current User” when running the tool .

  Lastly, ShowActivityVisitor performs a similar operation to the previous class but centers it around a certain activity. The query consists on given an activity name, show where the activity is located in terms of pool, lane, and sub-lane, and the actors that are assigned to his business functions. For this query to execute, the user logged in has to provide an activity name. Then, the Manager class requests Atlas to retrieve all the actors and proceeds to call the class ShowActivityVisitor. The class then searches for the activity in the loaded business process and uses the permission from the lane, where the activity is located to cross the information with the actors retrieved by Atlas. After this, an output is presented containing all the actors and the business functions that they can execute in that activity. To obtain the output of given by this class, the user has to select the option ”Show Actors of Activities” when running the tool.

To have a better representation of how the tool works and the steps done during his execution, the appendix B shows the sequence flow diagram starting with the actor’s login and ending with the execution of the ShowRelevantProcessVisitor query. Some parts of the sequence diagram are simplified since the objective was to create a better understanding of the interactions between the users with the tool, and the interactions between the classes Manager, Mapper, BusinessProcessChecker, and queries. These simplified parts are: ”Map element”, ”Check Business Process”, and ”Visit”.

”Map element” substitutes the reading of the JSON sent by Atlas for a given process element and the creation of the same element in the tool. The part is not equal for every business process element. For instance, elements such as data objects, and data stores just need to have an object created, and to be inserted in a Java Collection of the corresponding pool. On the other hand, lanes need to go through a recursion function, if the information in the JSON indicates that they have sub-lanes. This function will be executed until there are no more sub-lanes inside sub-lanes. ”Check Business Process” substitutes the process of seeing if there are faults or warnings in the business process. ”Visit” substitutes the execution of all the visit methods defined for every business process element. This execution produces the output for the query that then will be shown to the user.

5

Case Study, Demonstration & Results

Contents

This chapter aims to present the case study, Rental Cars-R-Us, which focus on the business process *Rent-a-Car*. Then, section 5.2 does a demonstration on how to create classes and objects using the Atlas project and applies the solution to the case study's process, showing the results of the queries for a given actor of the case study's business process.

5.1 Case Study: Rental Cars-R-Us

This section summarizes and goes through the case study of the *Chapter 14: Rental Cars-R-Us case study* of Paul Harmnon's book, *Business Process Change A Business Process Management Guide for Managers and Process Professionals*, in his fourth edition. In this chapter, the author presents a company with some business process problems and offers a solution in the form of a process redesign. To have a completer business process and more adapted to the problems that this thesis aims to correct, the process was adapted to better fit the example, and some elements of the book *Enterprise Ontology A Human-Centric Approach to Understanding the Essence of Organisation* by Jan Dietz and Hans Mulder, were also used.

Rental Cars-R-Us is a small company established in Vancouver, Canada. The main objective of the company is to rent cars to its customers. In the last years, the company has been acquiring other car rental companies, making the company grow larger. Due to the increment, the company has suffered some quality and consistency losses, which have led to the need for process redesign of the core process, *Rent-a-Car*.

Rental Cars-R-Us rents cars to their customers by booking them using the phone. The cars that Rental Cars-R-Us offers are organized in groups that depend on the rate charged and deposit amount. When a customer desires to rent a car, the rental must ask for the documentation of the client. Since the company allows to have a deposit payer, invoice payer, and a driver that can be different people than the renter, it is also needed their documentation [36]. Then, it will be necessary to specify the start and end date of the rental, as well as the pick-up branch, return branch, and the credit card for the payments. Once the desire car group and car are selected, the deposit price and the base price of the rent are calculated, and it is asked for the client to confirm the rent.

The Rental Cars-R-Us organization is composed of the Headquarters of the company; operating companies that exist for each country where the rental has a business; local area franchises; service depots that are in charge of maintaining every car of a local franchise; and branches that can be of three categories: airport, city, and agency. Figure 5.1, was extracted of [37] and shows the structure of the company.

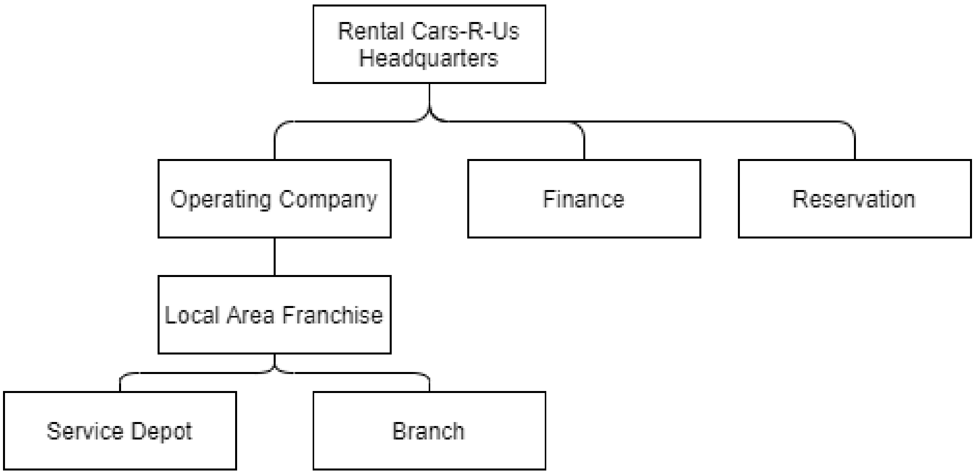


Figure 5.1: Rental Cars-R-Us organization

5.1.1 Rent-a-Car Process

A process redesign team studied the local area franchise in Calgary, Alberta, Canada for the process improvement, as requested by the company's Chief Operating Officer (COO). Intending to have a more concrete understanding of the problems of the process, the redesign team centered its focus on one of the biggest branches of the company, the airport branch of the Calgary franchise, which gets many complaints.

After visiting the branch, the team designed figure 5.2, which represents the structure of the management organization of the company's branch offices. When designing the process, the team established that the process began when a client requested a car and concluded when the client returned the car and paid for the rental. Then, they interviewed the managers and employees of the branch and developed a scope diagram of the *Rent-a-Car* process. In the development of the scope diagram, the team took into consideration individuals and other organizations that interacted with the Rent-a-Car process, as well as other processes and systems that interacted with it. They also studied what were the inputs and outputs of the process, plus the nature that they took (e.g., telephone calls, reports). Lastly, it was also considered policies issued by the headquarters; rules in employee manuals; and other legal requirements that constraint the process, together with other resources, such as databases and software applications.

Figure 5.3 is the final design of the scope diagram and is used to provide the redesign team with an overview of the process. In the figure, the process's **inputs** are rental inquiries (which corresponds to the requirements that the clients want in their reservations); agreement's confirmation; driver's license (presented when picking up the car); the car (returned when the rental ends); car rent payments; complaints in case the customer found something wrong (either with the rented car, the agreement, or any other not pleasant experience); the client's credit card approval (provided by a credit card processing center), and if the client was previously a client of the company, client information. During and after the process execution, the **outputs** produced are the rental agreement with the customer; the rented car prepared for the customer to use; the customer information given by the customer for optimizing other processes; a report of the process performance to improve the process, and a request to a credit card center to obtain a client's credit card approval. To work with these inputs and generate the outputs the process has **employees** who take care of the client reservations and everything related to the car. To save the agreements with the clients, their information, and the history of clients, the company has a **database**. Lastly, the company has **policies** and other guidelines that define it, which help steer the business of the organization. As can be seen, there are some problems in certain parts of the process, which will be explained in section 5.1.2.

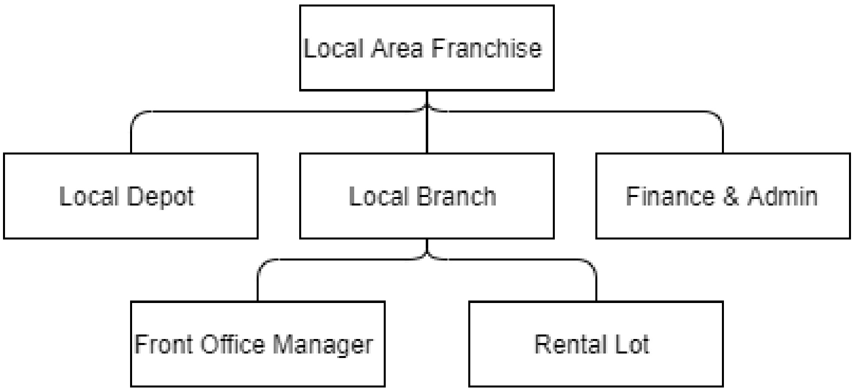


Figure 5.2: Rental Cars-R-Us organization in franchise level

With the scope diagram designed and knowing how the company was structured both at the high level and the branch level, the redesign team was able to re-create the As-Is business process of the *Rent-a-Car* process using the BPMN provided by the bpmn.io that Atlas uses for modeling. Figure 5.4, shows the *Rent-a-Car* based on the combination of [36], [37], and the adaptation made.

The process starts with a request by a client to rent a car. The headquarters receive this information and ask for the client's identification and other requirements needed (start and end of the rent, location of pick up and return car group, credit card information). Then, after selecting the desired car, the headquarters check for the availability of the vehicle, using the rental database of the company. If the car is available, the prices (deposit and base) are calculated, and it is asked for the client to confirm the reservation. Once the client confirmation is obtained, a record is created in a Rental database with the contract.

One of the company's policies is said that a full deposit has to be paid until the start of the rental. Therefore, when the driver comes to pick the car up on the first day of the rental the Finance department

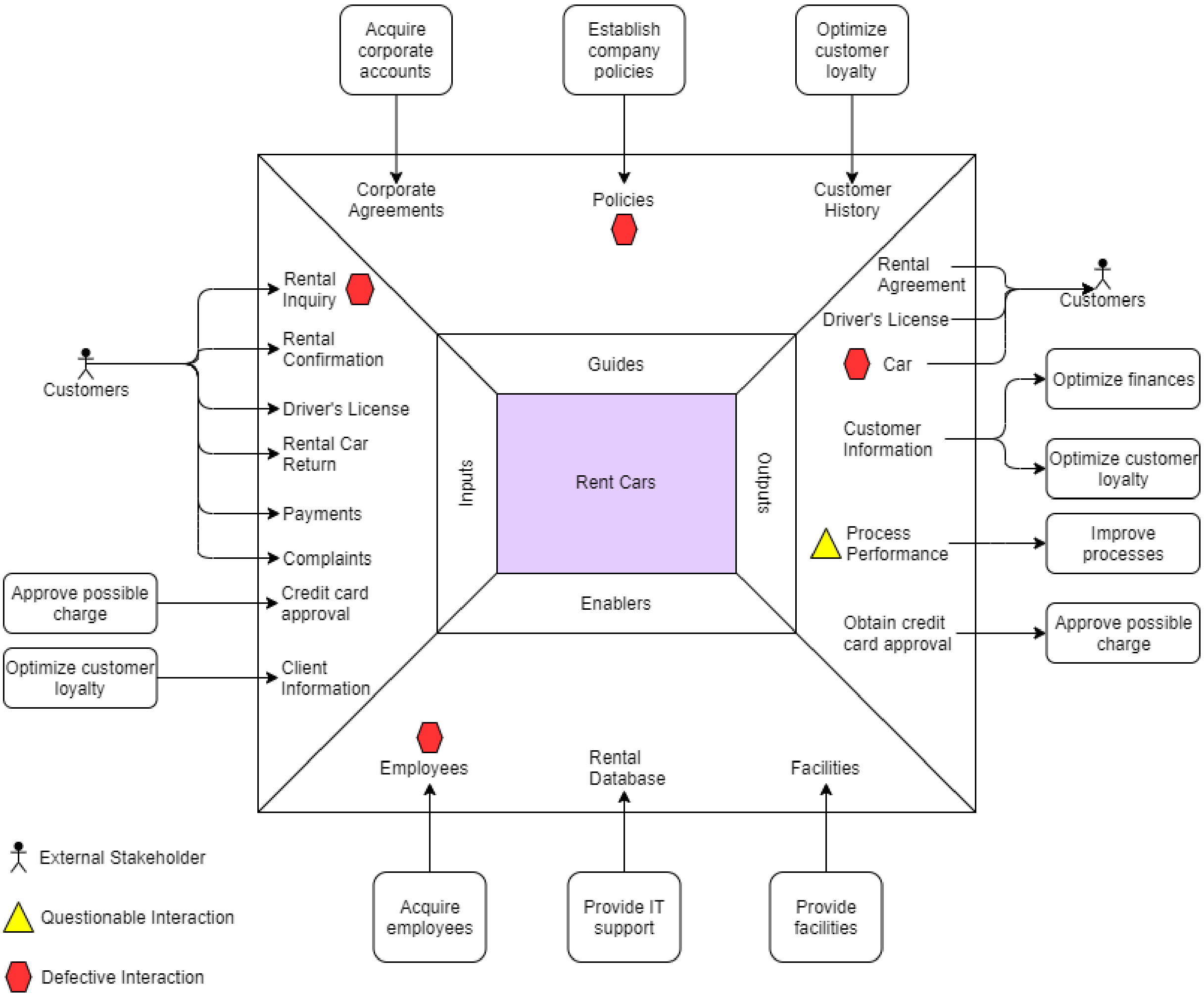


Figure 5.3: Rent-a-Car Scope Diagram

needs to check for the deposit payment. If the deposit has been paid, the Depot is in charge of preparing the car. However, if the rented vehicle is not in the Depot, a higher group car is selected, and the client does not need to pay extra. When the car is moved to the Branch Office's Front Office, the driver's license of the person that comes to pick up the car is checked to see if it is valid. In case of a valid driver's license, the vehicle is given.

When the car is returned, the Rental Lot checks for car damage and gas and makes notes on the contract in case some fee is needed to be paid. A fee is also paid if there was a delay in the return of the car, or if the car was returned in another branch than the established on the contract. Either way, once the car is returned, it is moved to the Depot ready for maintenance, where it will be cleaned, the tank will be filled, and if there is any damage, it will be repaired. With the maintenance completed, the Depot inserts what was done to the car in the rental database and the vehicle is ready to be prepared for the next reservation. At the same time the maintenance is being done in the depot, the headquarters are in charge of the payment process. Once this is concluded, the process *Rent-a-Car* ends.

5.1.2 Problems

”Several problems were uncovered” [37], during the meetings and interviews with managers and employees of the Calgary branch that later were put in the scope diagram. These problems were such as, ”policies were unclear or confusing, and thus clerks taking reservations on the telephone often made mistakes in completing the reservation” [37]. However, once the client came to pick up the car, many of the mistakes were often corrected. Nevertheless, ”customers still complained about the time spent” [37]. This situation led to the headquarters legal staff to send complaints to the Reservation management ”about the incorrect the reservations that put company insurance at risk”. To which the reservation's staff said it was because of the unclear policies that the headquarters had.

Other problems occurred during the set up of the cars and ” A car might not have a global positioning system as ordered, or a car might be logged into the wrong slot on the lot, so a customer could not find it” [37]. The other problems, that led to the complaint of clients, occurred during the maintenance of the cars. For instance, customers found paper cups in the back seat area or that the tank was not full. ”The depot manager blamed the problems on poor training of the employees who carry out auto maintenance and preparation” [37].

5.1.3 Redesign Solution

At this point, the redesign team had three main problems to focus on, in the business process. Those were:

1. ”The problem customers and the organization had getting the reservation agreement right” [37].
2. ”The problem the organization had getting new cars prepared as requested” [37].
3. ”The problem that resulted from managers not being on top of what was happening and responding quickly enough” [37].

The solutions proposed in the *Chapter 14: Rental Cars-R-Us case study* consist, inter alia, of revise Ethe rental agreement to make it easier and less ambiguous”; create Ea website where customers could make their own reservations”; retrained ”depot personnel in the preparation of cars” and develop ”a preparation quality checklist and requiring managers to check each car before placing it in a stall”.

From these solutions proposed in [37], this thesis will apply the solution presented in Chapter 4 to provide the quality checklists for all the employees of a process to check the work that they are assigned and need to do. What is proposed in this thesis is to apply an access control models to create a clear

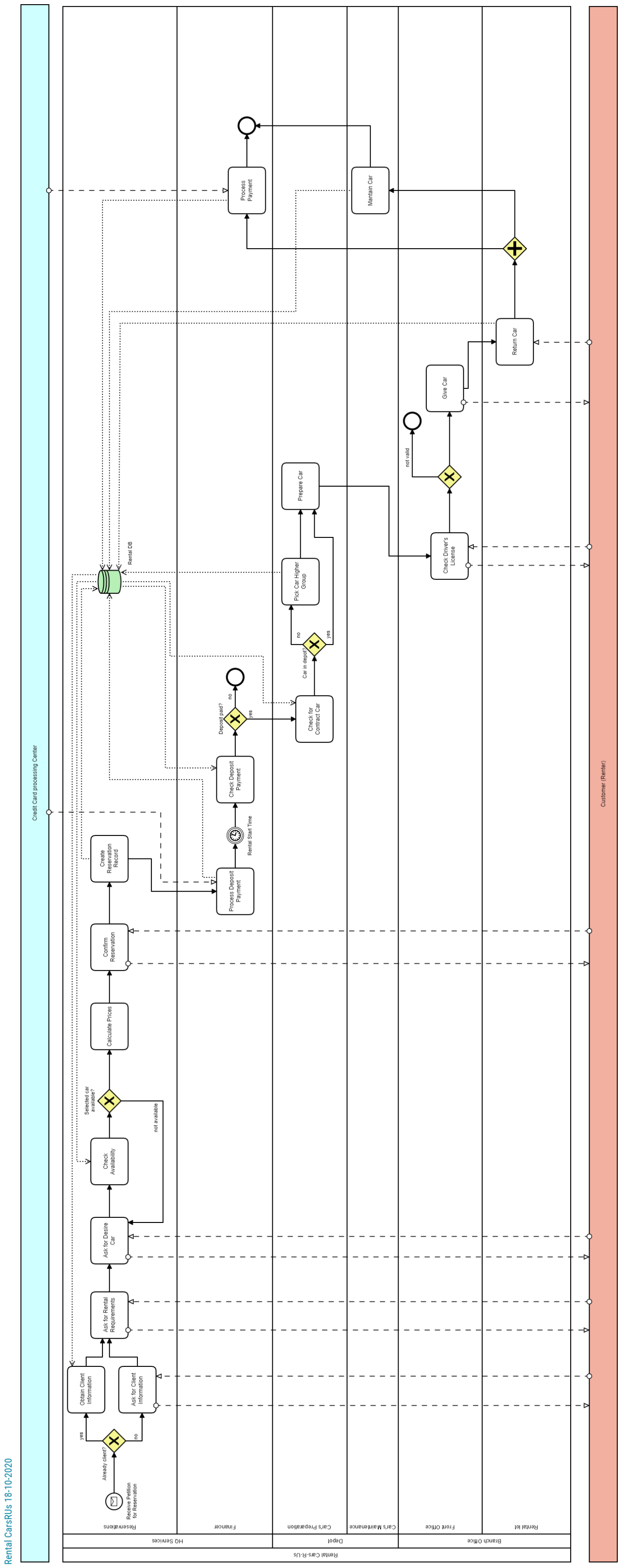


Figure 5.4: As-Is Rent-a-Car business process

direction of the tasks that are relevant to the employees. By integrating access control models to a notation, such as BPMN, every time the process changes in terms of 'who can do what', and the tasks assigned to the people change, the list of tasks done by them will be updated, and a new task quality checklist is presented.

Since this work uses Atlas for the design of business processes, the only requirement needed is for the Rental Cars-R-Us organization to have an account on Atlas. With it, they will be able to view the process, change it, and use the tool that this work implemented. The following section will go through the process of designing the Rent-a-Car process in the Atlas project and showing the output of the tool when used with the Rent-a-Car process, with the objective of correcting the problems in relation to set up and maintenance of the cars.

5.2 Demonstration & Results

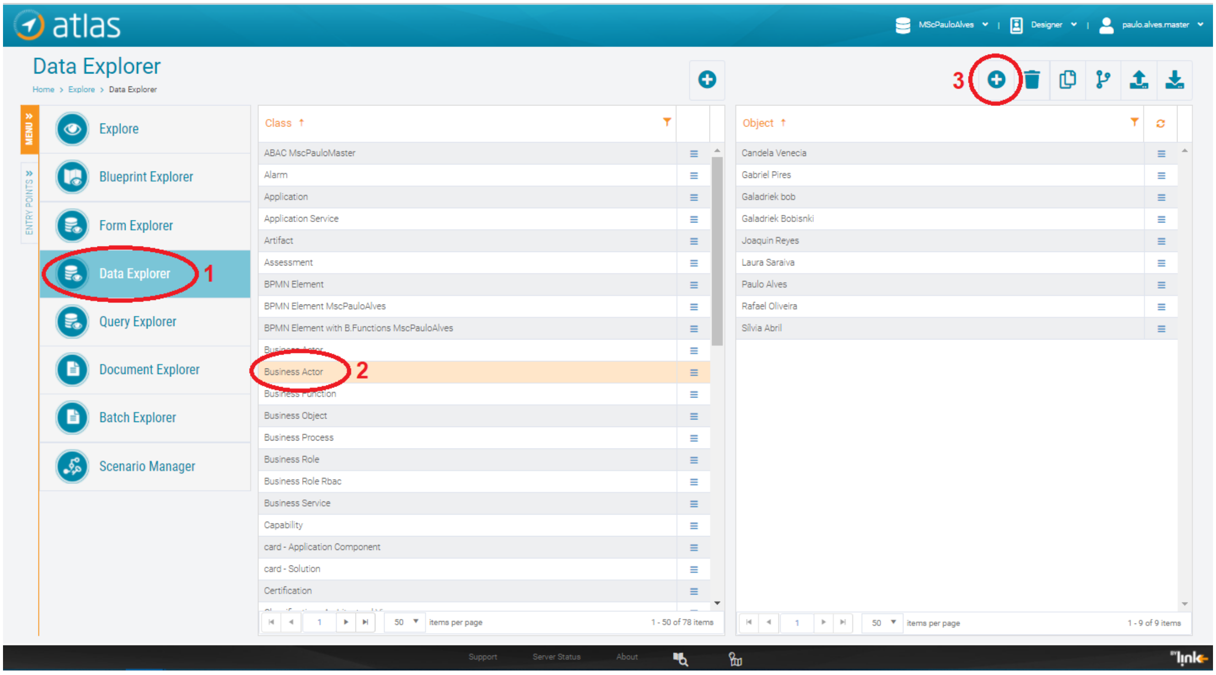
The content of this section will be split into two parts. The sub-section 5.2.1 will go through every step needed to design the Rent-a-Car business process of the case study, presented in the previous section, with the access control models using the Atlas project. It will cover the creation of the actors of the company, the creation of the activities with business functions, and the creation of access control models. Then, on sub-section 5.2.2, it will be shown how the developed work operates, and the results that the queries return will be explained.

5.2.1 Process Design using Atlas

5.2.1.A Creating Actors

Since the solution depends highly on the actors and their attributes and roles, the process's actors are needed to be created for the correct solution behavior. For that matter, the first thing to do is to add all the actors involved in the process. However, the case study does not mention any particular actors, attributes, and roles (aside from the departments of the company). Therefore, the actors and their properties presented in this sub-section were made up of the matter of the example.

In Atlas, an actor is just an instance of a class. The place in Atlas where the classes and objects are created is the "Data Explorer", selectable right after the login. Once there, the next step is just to select the class "Business Actor" and click on "Add Object". Figure 5.5 shows the interface when selecting the "Data Explorer" option and indicates the three steps necessary to create an object. Then, a form will appear where the attributes of the actor that is being created will have to be written. For instance, in figure 5.6, the form is filled with the attributes of Paulo Alves who works for the "Depot" division; with the



Class
ABAC MscPauloMaster
Alarm
Application
Application Service
Artifact
Assessment
BPMN Element
BPMN Element MscPauloAlves
BPMN Element with B.Functions MscPauloAlves
Business Actor
Business Actor
Business Function
Business Object
Business Process
Business Role
Business Role Rbac
Business Service
Capability
card - Application Component
card - Solution
Certification

50 items per page 1 - 50 of 78 items

Object
Candela Venecia
Gabriel Pires
Galadriek bob
Galadriek Bobisnki
Joaquin Reyes
Laura Saraiva
Paulo Alves
Rafael Oliveira
Silvia Abril

50 items per page 1 - 9 of 9 items

Figure 5.5: Create Objects of Class ”Business Actor”

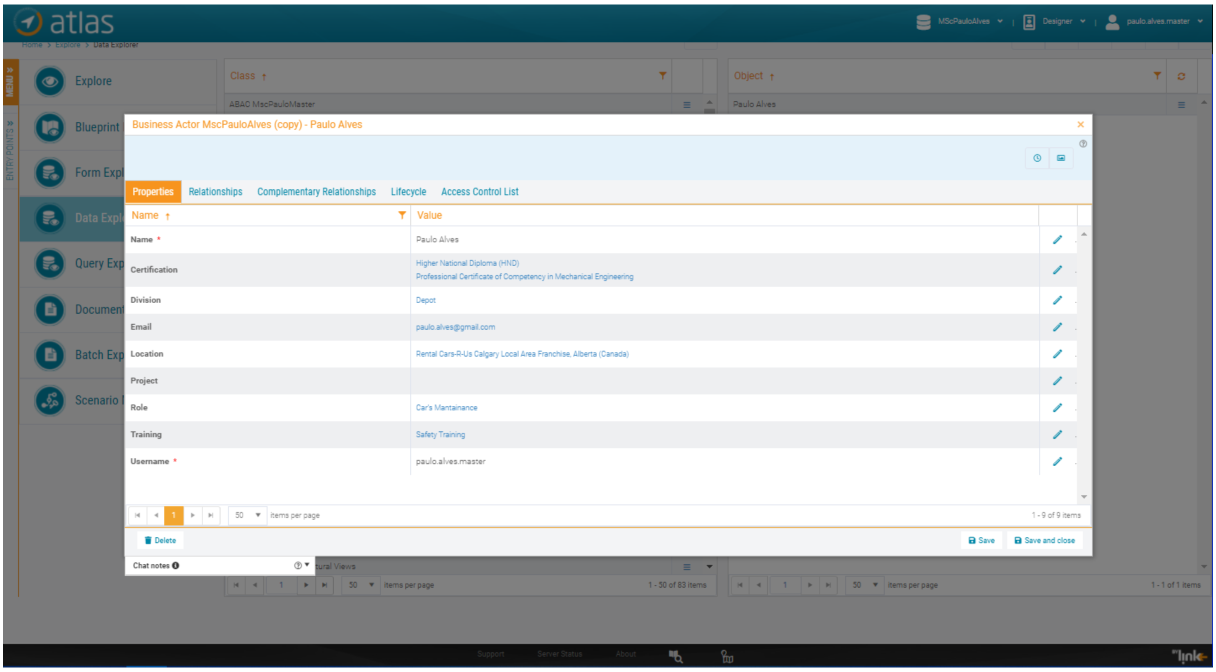
role ”Car’s Maintenance”; in the ”Rental Cars-R-Us Calgary Local Area Franchise, Alberta (Canada)”; has the email ”paulo.alves@gmail.com”; the Atlas’s username ”paulo.alves.master” (this username is the the one used for the login in Atlas and in the tool); has the ”Higher National Diploma (HND)” and the ”Professional Certificate of Competency in Mechanical Engineering”; and has had training in ”Safety”.

For the case study presented in the previous chapter, Atlas was populated with the rest of the Rental Cars-R-Us personal. In the ”Rental Cars-R-Us Headquarters, Vancouver (Canada)”, three more people were added. One to the ”Finances” division, and two people to the ”Reservation” division. In the Calgary Airport branch, which belongs to the ”Rental Cars-R-Us Calgary Local Area Franchise, Alberta (Canada)”, six people were added. Three of them to the ”Depot” division, one to the ”Calgary Airport Front Office” and the last two to the ”Calgary Airport Rental Lot” division.

5.2.1.B Creating Activities and Designing Business Process

After adding the company’s actors to Atlas, the next step will be the creation of activities and their business functions. This task can be done in two different ways:

The first option is almost the same as the creation of the actors. Equally as actors, in Atlas, activities are also objects, but from the class ”BPMNElement”; Therefore, to create an activity the process can be the same as the shown in figure 5.5, but instead of selecting the ”Business Actor” class in step 2, the selection is going to be ”BPMNElement”. Then, similarly to the creating process of a new actor, a form



Business Actor MscPauloAlves (copy) - Paulo Alves

Name	Value
Name *	Paulo Alves
Certification	Higher National Diploma (HND) Professional Certificate of Competency in Mechanical Engineering
Division	Depot
Email	paulo.alves@gmail.com
Location	Rental Cars-R-Us Calgary Local Area Franchise, Alberta (Canada)
Project	
Role	Car's Mantainance
Training	Safety Training
Username *	paulo.alves.master

1 - 9 of 9 items

Figure 5.6: Actor's form to write object's properties

will appear where all the properties of the activity are going to be written. Figure 5.7 shows the form of the activity ”Ask for Client Information”. Despite having so many fields, the only ones needed for the tool to work are: ”Business Functions”; ”Read” which refers to the Data Stores from which the activity reads data; ”Write” which refers to the Data Stores to which the activity writes data; ”Output” which refers to the Data Objects that the activity has as an output, and lastly ”Input” which refers to the Data Objects that the activity has as an input.

The other option for creating activities is more graphical. To use this option, first, it would be needed to select ”Blueprint Explorer” in the Atlas menu and then, in the explorer, select the option with the name ”BPMN” which is the editor for BPMN 2.0. Figure 5.8 shows the interface if the ”Blueprint Explorer” menu with the ”BPMN” option selected. With the BPMN editor open, the business process can be designed. For every element put on the editor, an object is created, and his attributes can be edit by choosing the ”Detail” option, as shown in figure 5.9, which will open the BPMNElement's form shown previously in figure 5.7.

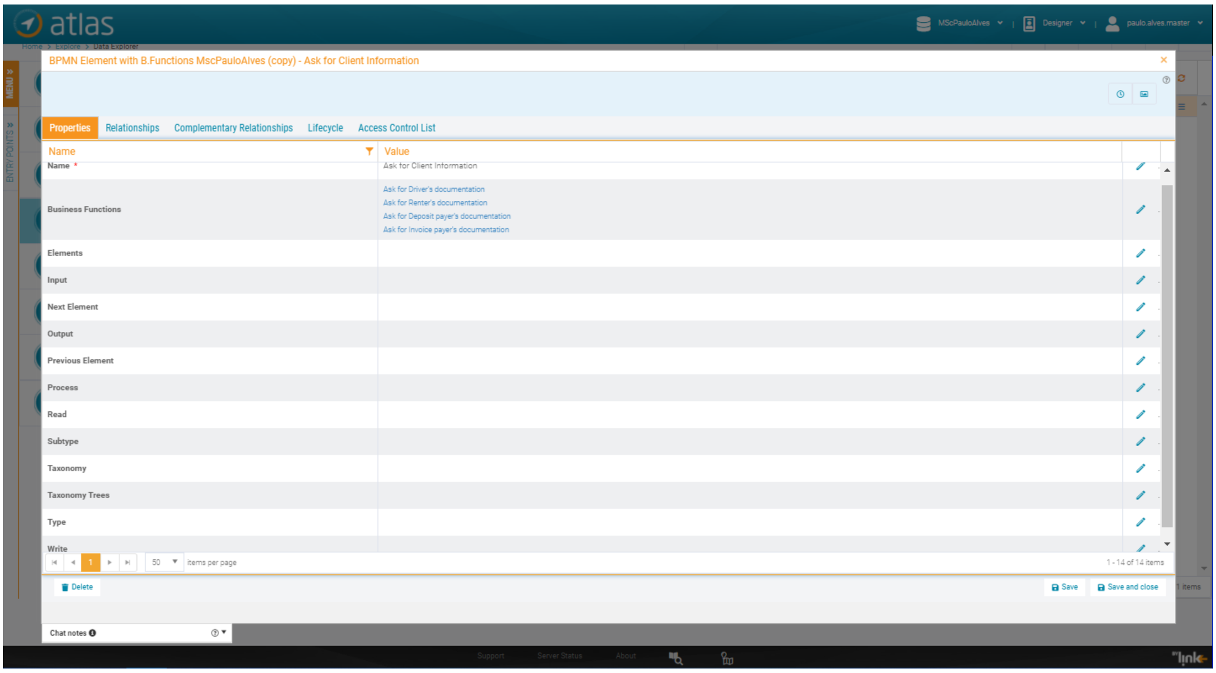
These two options not only work for the process activities, but for every element that can be put on the process that is mapped to a class. For instance, for pools, that are also mapped to a class, the properties presented in their form are, the data elements owned (i.e., data elements modeled inside the pool) and the lanes that the pool contains.

5.2.1.C Creating and Adding Access Control Models

For Atlas, a lane is just another class. Therefore, to instantiate an object of this class the process used is the same used in the previous sub-section to create BPMNElements. Consequently, there will be a lane form, that is going to be composed of the BPMNElements of the lane (i.e., activities), possible Sub-lanes, and lastly, the permissions that the lane has that will define the model that the lane is (RBAC or ABAC).

In the Rental Cars-R-Us case study, there is no explanation of the policies and who can access what resources. Hence, the process's access control models were made to better fit the example, and to show how to design in Atlas and how the tool works. The results of the tool's queries depend on two main variables. Those are, the properties that the actors have, and the permissions associated with each lane. This means that, to a same set of actors, if the permissions of a lane vary, the outputs returned could be different in the tool's queries and vice-versa. For that matter, the policies for the example are explained in the next paragraphs, where is being associated to each lane, the attributes or roles necessary for those tasks.

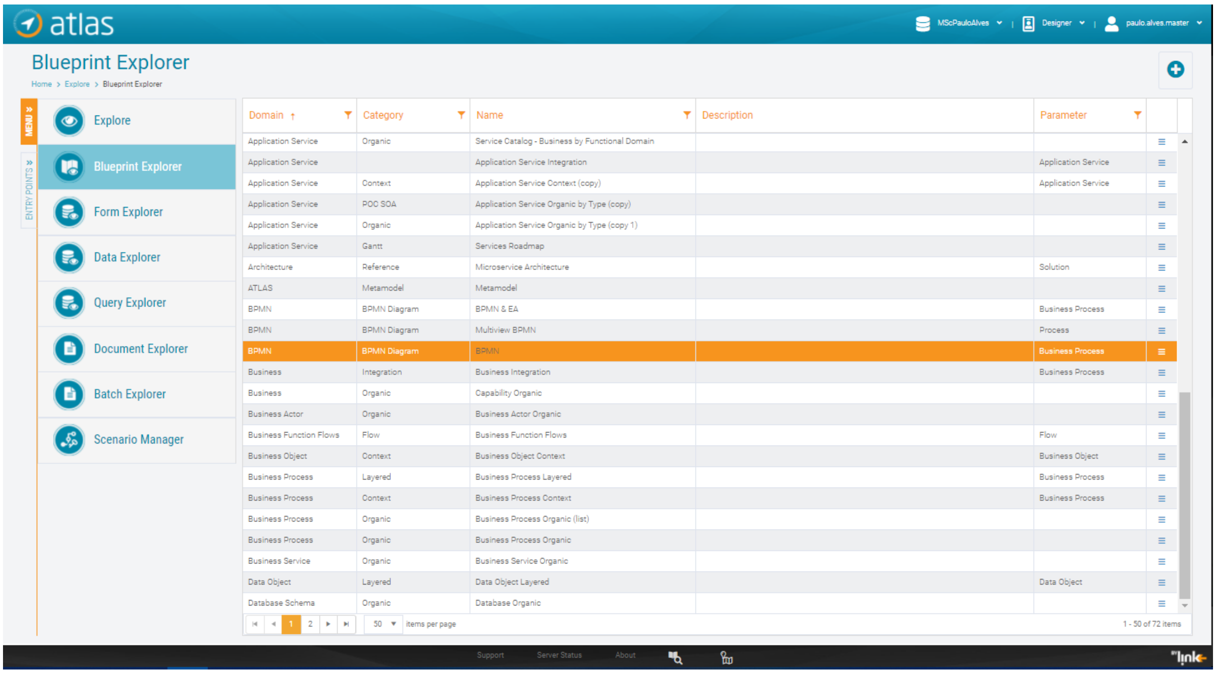
The finance operations of this process, are carried out by people that belong to the Finance division, with the location in the Headquarters of the company, and have the role of bookkeeping. On the other sub-lane of the HQ Service, to carry out the business functions of the Reservations Department, it is needed to be on the Reservation division and to work on the Headquarters of the company.



BPMN Element with B.Functions MscPauloAlves (copy) - Ask for Client Information

Name	Value
Name *	Ask for Client Information
Business Functions	Ask for Driver's documentation Ask for Renter's documentation Ask for Deposit payer's documentation Ask for Invoice payer's documentation
Elements	
Input	
Next Element	
Output	
Previous Element	
Process	
Read	
Subtype	
Taxonomy	
Taxonomy Trees	
Type	
Write	

Figure 5.7: BPMNElement's form to write object's properties



Domain ↑	Category	Name	Description	Parameter
Application Service	Organic	Service Catalog - Business by Functional Domain		
Application Service		Application Service Integration		Application Service
Application Service	Context	Application Service Context (copy)		Application Service
Application Service	POC SOA	Application Service Organic by Type (copy)		
Application Service	Organic	Application Service Organic by Type (copy 1)		
Application Service	Gantt	Services Roadmap		
Architecture	Reference	Microservice Architecture		Solution
ATLAS	Metamodel	Metamodel		
BPMN	BPMN Diagram	BPMN & EA		Business Process
BPMN	BPMN Diagram	Multiview BPMN		Process
BPMN	BPMN Diagram	BPMN		Business Process
Business	Integration	Business Integration		Business Process
Business	Organic	Capability Organic		
Business Actor	Organic	Business Actor Organic		
Business Function Flows	Flow	Business Function Flows		Flow
Business Object	Context	Business Object Context		Business Object
Business Process	Layered	Business Process Layered		Business Process
Business Process	Context	Business Process Context		Business Process
Business Process	Organic	Business Process Organic (list)		
Business Process	Organic	Business Process Organic		
Business Service	Organic	Business Service Organic		
Data Object	Layered	Data Object Layered		Data Object
Database Schema	Organic	Database Organic		

1 2 | 50 items per page | 1 - 50 of 72 items

Figure 5.8: Blueprint Explorer menu interface

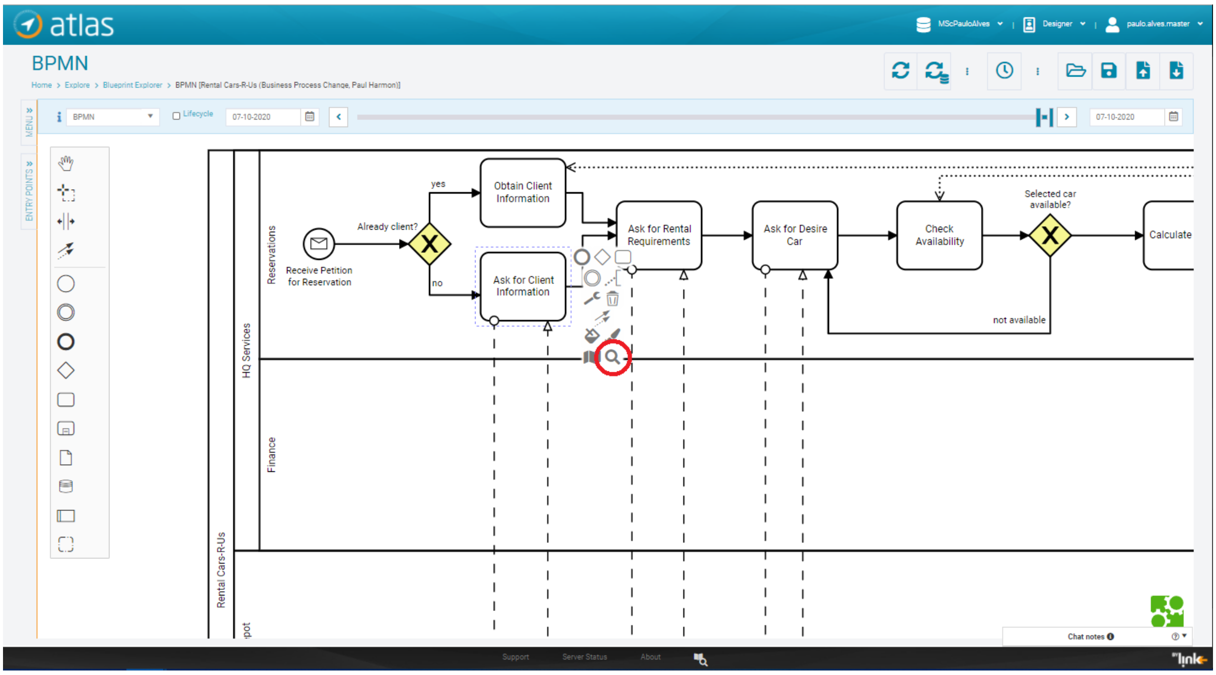


Figure 5.9: Bpmn.io editor

In the Local Area Depot, there are two teams, each of them with their business functions. One of them is dedicated to the cars' preparation. The workers that execute these business functions need to be on the Depot division of the Calgary Local Area Franchise and have the Car' Preparation role. On the other hand, the other team is focused on the maintenance of the cars. To execute these business functions, the workers must have the Car' Maintenance role, and equally to the other team, have to work for the Depot of the Calgary Local Area Franchise.

At a branch level, the Branch Office Manager is in charge of overseeing and can also execute the tasks both the Front Office and the Rental Lot. Therefore, when modeling the process the parent-lane ”Branch Office” is of type RBAC, and the role included in the permission is ”Branch Office Manager”. In the sub-lane ”Front Office”, the policy states that to execute the tasks it is needed to belong to the Calgary Airport Front Office of the Rental Cars-R-Us Calgary Local Area Franchise of Alberta. Similarly, the Rental Lot policy states that it's needed to be part of the Calgary Airport Rental Lot of the Rental Cars-R-Us Calgary Local Area Franchise of Alberta.

Figure 5.10 shows the access control models' application, with their permissions and business functions, that satisfy what has been described. Since for the execution of the queries the actor used belongs to the Car's Maintenance team, there is a specification of the business functions of both of the sub-lanes of the Depot.

5.2.2 Tool Use and Understanding

In this sub-section, the results of the different queries will be shown and explain. Since failures can occur in the mapping and the modeling of the business process, the sub-sub-section 5.2.2.A will be center on displaying different cases, and the sub-sub-section 5.2.2.B will present the result of the queries when the business process is correctly model regarding the expansion.

To start using the tool, first the java program has to start running. Once this is done, it will be asked to the user to do the login using the Atlas's credentials (i.e., the username and password used to login to the Atlas project). After the credentials are classified as valid, the tool asks the user to insert the name of the Atlas's repository and the name of the business process that the user wants to load to the tool. Then, if the repository and the process exist, the tool presents a number of options to the users. These are, the queries and an option to change the business process loaded in the tool, which will ask the user to indicate the Atlas's repository and the business process name again, to load the new process to the tool. Figure B.1 shows the sequence diagram and the interactions of the user with the tool for the login commands and the loading of the business process (if the repository and the business process exist).

5.2.2.A Failure & Warning Cases

When modeling the business process and creating everything it needs, mistakes can happen if not all the constraints are satisfied. As explained in chapter 4, the tool developed contains a class dedicated to checking if the process was model correctly, and in case of fault, then the queries will not be executable. However, this will not happen in case of a warning.

There are two types of possible failures. The first type is related to mapping errors. These occur when a certain component of the Atlas' modeled business process is missing. For instance, if the process is empty or does not exist. In these cases, an exception in the tool is thrown, indicating the occurrence of a mapping error. The other type is related to the constraints of having activities inside of parent-lanes, using data objects/stores not owned by the pool where they are being used, having business functions with no permission associated, and more. Figure 5.11 shows a capture of the developed tool's interface

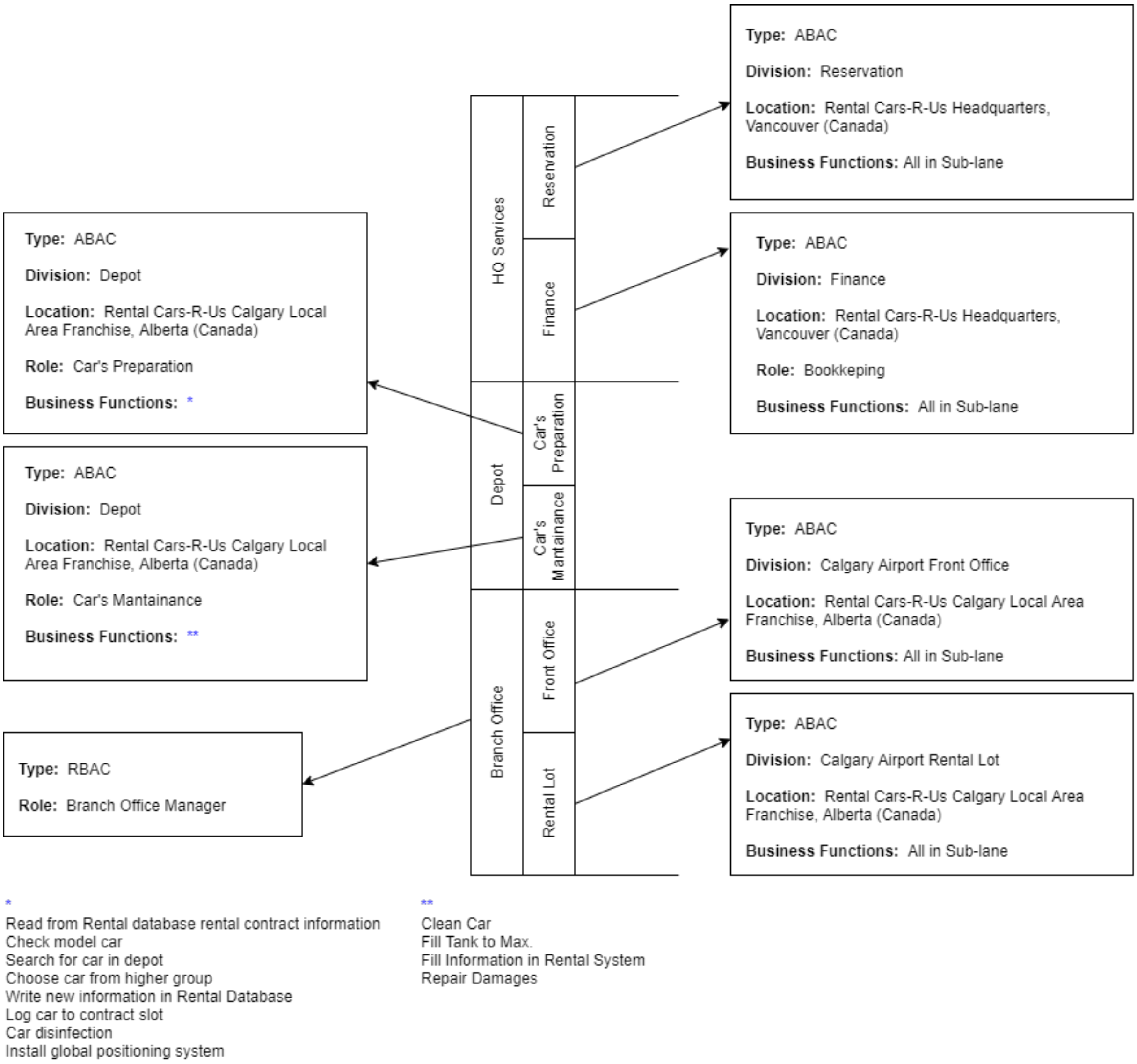

Type: ABAC
Division: Reservation
Location: Rental Cars-R-Us Headquarters, Vancouver (Canada)
Business Functions: All in Sub-lane
Type: ABAC
Division: Finance
Location: Rental Cars-R-Us Headquarters, Vancouver (Canada)
Role: Bookkeping
Business Functions: All in Sub-lane
Type: ABAC
Division: Depot
Location: Rental Cars-R-Us Calgary Local Area Franchise, Alberta (Canada)
Role: Car's Preparation
Business Functions: *
Type: ABAC
Division: Depot
Location: Rental Cars-R-Us Calgary Local Area Franchise, Alberta (Canada)
Role: Car's Mantainance
Business Functions: **
Type: RBAC
Role: Branch Office Manager
HQ Services
Reservation
Finance
Depot
Car's Preparation
Car's Mantainance
Branch Office
Front Office
Rental Lot
Type: ABAC
Division: Calgary Airport Front Office
Location: Rental Cars-R-Us Calgary Local Area Franchise, Alberta (Canada)
Business Functions: All in Sub-lane
Type: ABAC
Division: Calgary Airport Rental Lot
Location: Rental Cars-R-Us Calgary Local Area Franchise, Alberta (Canada)
Business Functions: All in Sub-lane
*
Read from Rental database rental contract information
Check model car
Search for car in depot
Choose car from higher group
Write new information in Rental Database
Log car to contract slot
Car disinfection
Install global positioning system
**
Clean Car
Fill Tank to Max.
Fill Information in Rental System
Repair Damages

Figure 5.10: Access Control Model's Diagram

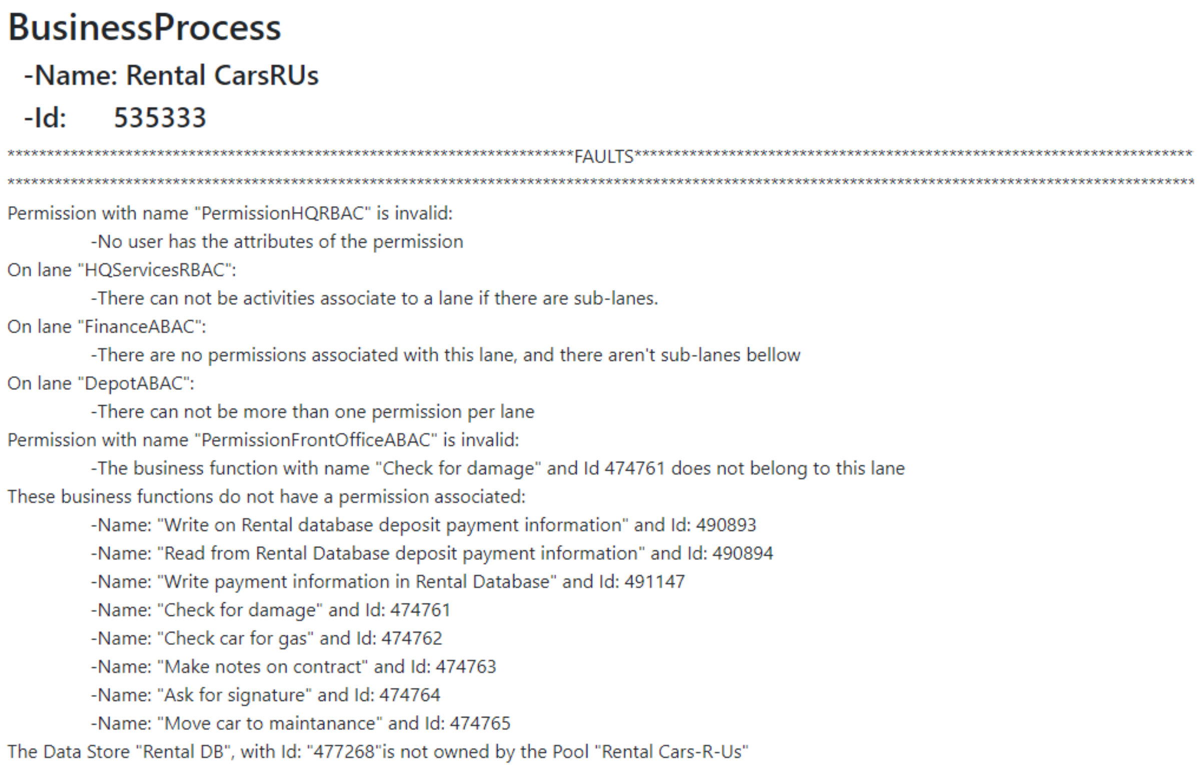
if this type occurs in the *Rent-a-Car* business process.

The ”There can not be more than one permission per lane” failure happens because there are two or more permissions associated with the same lane, which contradicts the idea of a separation of responsibilities between lanes, since there is a lane where different business functions are assign to different policies. In the particular case of the image, the Depot lane was model without sub-lanes (Car’s Preparation and Car’s Maintenance). Meaning that the only way to split the assignments of the actors of the cars’ preparation team, from the cars’ maintenance team, was to add the two permissions to the Depot’s lane, which is not allowed because it does not respect the split responsibilities lane idea. To correct this situation, the business process should be modelled as presented in figure 5.10.

The ”These business functions do not have a permission associated” failure exists because there are business functions inside the activities that are not associated with a permission. Hence, nobody will execute them, causing problems and improper behavior of the business process. Similarly, another failure happens to a lane without sub-lanes that does not have a permission. This denotes that nobody has assigned the business functions of the lane. Lastly, in this point, ”No user has the attributes of the permission” is also considered a failure since this means that nobody will be able to execute the permission’s business functions due to no one having the attributes that the permission has. In terms of modeling to correct this, either the permission’s attributes are changed, or an actor is changed to fulfill the permission.

The other failure cases presented in figure 5.11 are related to objects missing or not needed when modeling the business process. The ”There can not be activities associated with a lane if there are sub-lanes” failure is caused because activities only should be associated with lanes with no more sub levels. The ”The business function with name ”Check for damage” and id 474761 does not belong to this lane” failure is caused by having business functions associated with lanes’ permissions where the business function’s activity does not belong. To correct it, the activity should be added to the lane where permission is, or the business function should be deleted from the permission. Finally, the remaining failure happens since data elements need to be owned by a pool in BPMN 2.0. In this particular case, there are activities in the process that use the data store ”Rental DB” to obtain information about the client or write information about the contract, but in the Pool ”Rental Cars-R-Us” does not have the data store in his attributes when the object is created. The data store should be added to the ”Rental Cars-R-Us” pool in his form to correct this.

Lastly, the warning cases are related to the data element present of the process. They occur when the pool owns data elements that are not being used anywhere in the process. Warnings do not limit the execution of the queries, but they are shown every time a query is executed.



```
BusinessProcess
  -Name: Rental CarsRUs
  -Id:     535333
************************************************************************FAULTS************************************************************************
******************************************************************************************************************************************************
Permission with name "PermissionHQRBAC" is invalid:
            -No user has the attributes of the permission
On lane "HQServicesRBAC":
            -There can not be activities associate to a lane if there are sub-lanes.
On lane "FinanceABAC":
            -There are no permissions associated with this lane, and there aren't sub-lanes bellow
On lane "DepotABAC":
            -There can not be more than one permission per lane
Permission with name "PermissionFrontOfficeABAC" is invalid:
            -The business function with name "Check for damage" and Id 474761 does not belong to this lane
These business functions do not have a permission associated:
            -Name: "Write on Rental database deposit payment information" and Id: 490893
            -Name: "Read from Rental Database deposit payment information" and Id: 490894
            -Name: "Write payment information in Rental Database" and Id: 491147
            -Name: "Check for damage" and Id: 474761
            -Name: "Check car for gas" and Id: 474762
            -Name: "Make notes on contract" and Id: 474763
            -Name: "Ask for signature" and Id: 474764
            -Name: "Move car to maintanance" and Id: 474765
The Data Store "Rental DB", with Id: "477268"is not owned by the Pool "Rental Cars-R-Us"
```

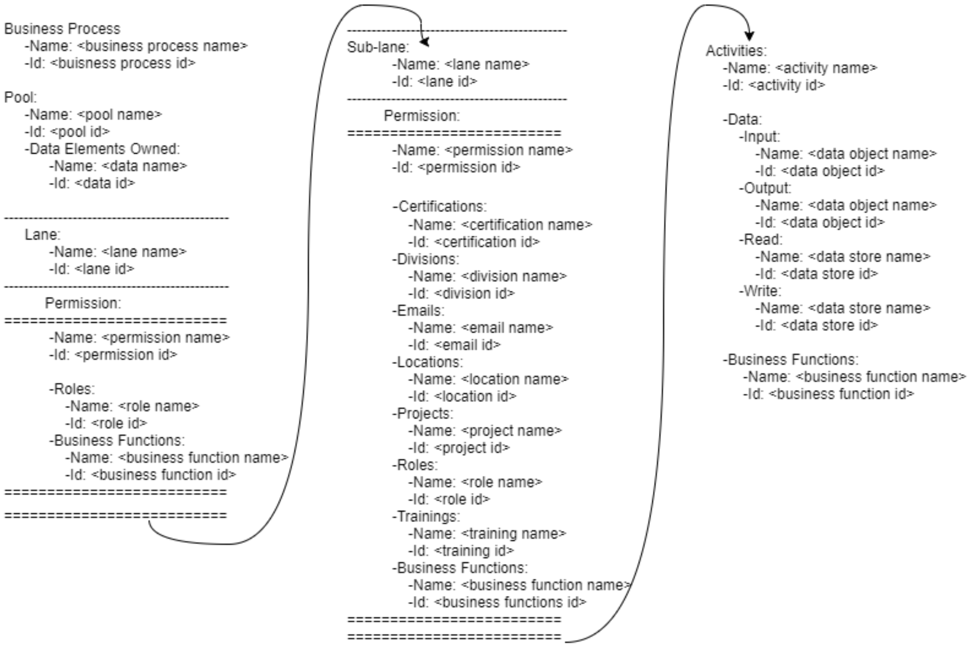
Figure 5.11: Failure cases' output

5.2.2.B Right Case

In the case that everything about the business process is correct, that is, every constraint presented in the previous section is fulfill, the software queries are available to execute. There are three of them in total, and these are:

- The "**Show Complete business process**" query shows the business process loaded into the software in a detail list view format. Figure 5.12 shows the format used. It starts with the pools of the business process, and then it descends the levels of the process until it reaches the business function's level.

- The "**Show Business Process For Current User**" query, shows only the parts of the process relevant to the user that is logged in. When the query is executed, it uses the user's information and the permissions that each Atlas's lane has. Then it crosses the information, presenting only the activities (with their business functions, data objects and data stores) that the user has access to execute. For instance, if the *Rent-a-Car* business process was loaded and the actor from figure 5.6 was logged in, the query would return the activities that belong to the sub-lane "Car's Maintenance" (see figure 5.13). This, because the actor belongs to the "Depot" division, in the location "Rental



```
Business Process
    -Name: <business process name>
    -Id: <buisness process id>

Pool:
    -Name: <pool name>
    -Id: <pool id>
    -Data Elements Owned:
        -Name: <data name>
        -Id: <data id>

----------------------------------------------
    Lane:
        -Name: <lane name>
        -Id: <lane id>
----------------------------------------------
        Permission:
=========================
        -Name: <permission name>
        -Id: <permission id>

        -Roles:
           -Name: <role name>
           -Id: <role id>
        -Business Functions:
           -Name: <business function name>
           -Id: <business function id>
=========================
=========================
```

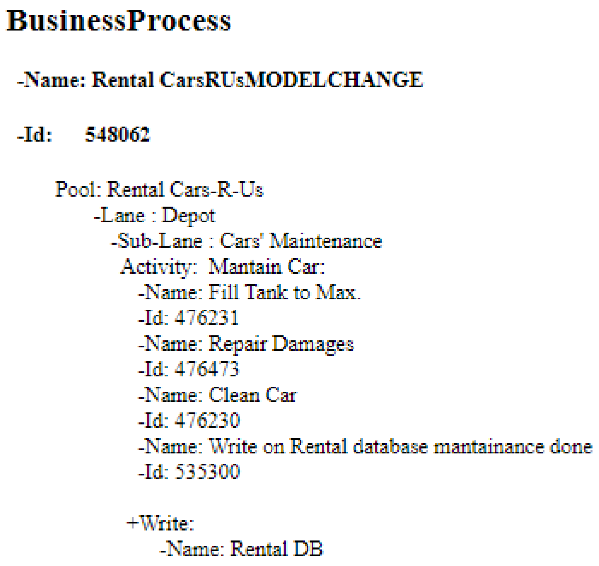
```
----------------------------------------------
Sub-lane:
        -Name: <lane name>
        -Id: <lane id>
----------------------------------------------
        Permission:
=========================
        -Name: <permission name>
        -Id: <permission id>

        -Certifications:
           -Name: <certification name>
           -Id: <certification id>
        -Divisions:
           -Name: <division name>
           -Id: <division id>
        -Emails:
           -Name: <email name>
           -Id: <email id>
        -Locations:
           -Name: <location name>
           -Id: <location id>
        -Projects:
           -Name: <project name>
           -Id: <project id>
        -Roles:
           -Name: <role name>
           -Id: <role id>
        -Trainings:
           -Name: <training name>
           -Id: <training id>
        -Business Functions:
           -Name: <business function name>
           -Id: <business functions id>
=========================
=========================
```

```
Activities:
   -Name: <activity name>
   -Id: <activity id>

   -Data:
      -Input:
         -Name: <data object name>
         -Id: <data object id>
      -Output:
         -Name: <data object name>
         -Id: <data object id>
      -Read:
         -Name: <data store name>
         -Id: <data store id>
      -Write:
         -Name: <data store name>
         -Id: <data store id>

   -Business Functions:
      -Name: <business function name>
      -Id: <business function id>
```

Figure 5.12: Output format of the ”Complete business process's” query

Cars-R-Us Calgary Local Area Franchise, Alberta (Canada)” and has the ”Car's Maintenance” role, thus, the permission of the lane is fulfilled and the access granted to the activities and the used resources by them. This query is helpful in situations such as the *Rent-a-Car* process, when the employees do not know what the should do because they do not have clear directions. This query presents a quality checklist that is always updated to the most recent business process loaded to the tool, and it's thought for the actors of the processes to know what was assigned to them in a given process.

The output of this query is also presented in a list view format, as seen in figure 5.15. The first part of the output is composed of the business process name and id (given by Atlas), followed by location in the business process (i.e., pool, lanes and sub-lanes) where the business functions are inserted in the permissions in Atlas. Lastly, it is presented the activity, the business functions and the data objects and data stores if the activity is associated to any. In figure 5.14 are presented different outputs for the different cases. In them, it is pretended to show how the different modeling of permissions affects the output of this query. For each of the cases, consider that for lanes Manager and Worker 1 whenever an arrow is pointing to ”business functions” there is a permission defined in the lane and the user logged in the tool has the required permissions. In the first case (left business process), there is no permission in the Manager lane, but there is on the Worker 1 lane, and the user has access to it. Therefore, the output shows the business functions BF1, BF2 and BF3 of activity Act1. In the second case (middle business process), the inheritance property



```
BusinessProcess

 -Name: Rental CarsRUsMODELCHANGE

 -Id:     548062

      Pool: Rental Cars-R-Us
           -Lane : Depot
             -Sub-Lane : Cars' Maintenance
              Activity:  Mantain Car:
                -Name: Fill Tank to Max.
                -Id: 476231
                -Name: Repair Damages
                -Id: 476473
                -Name: Clean Car
                -Id: 476230
                -Name: Write on Rental database mantainance done
                -Id: 535300

              +Write:
                   -Name: Rental DB
```

Figure 5.13: Query's result for "Paulo Alves" actor in *Rent-a-Car* business process

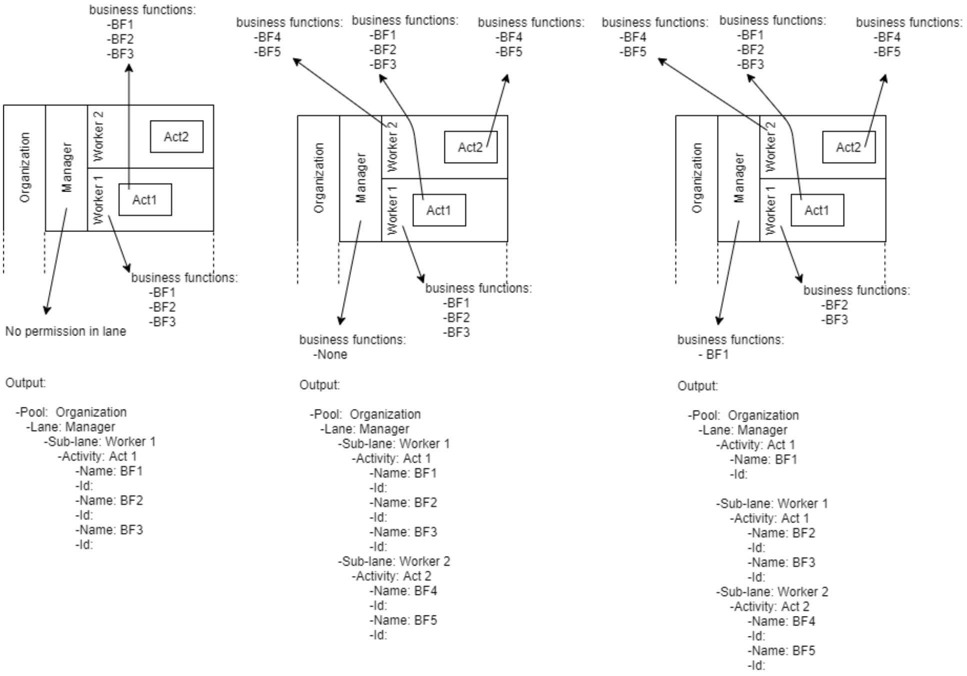
is applied. The user of the tool has the required permission of the lane Manager and Worker 1, thus he has accessibility to the business functions of the permission of the lane Worker 2 too (i.e., BF4 and BF5). Lastly, in the third case, the output changes because of the change of BF1. In this case this business function is only accessible to users that have the required permission for the Manager lane. Thus, in the output BF1 appears after lane Manager a not after lane Worker 1. Equally to the previous case the inheritance property is also applied. This type of modeling is useful for situations where a particular set of business functions need to be done by a higher rank permission.

To conclude, if the user does not have any of the required permissions of the business process loaded in the tool, the output is "You do not have any permissions in this business process". Therefore , the user is not in charge of anything related to the loaded business process.

- The "**Show Actors of Activities**" query allows the user to search for activities to know the pool and lane where it is and the available actors that can execute his business functions. This query will be helpful when a determined activity is badly being executed. In these cases, there is a need to know as quickly as possible who are the actors assigned to each of the business functions of the activity. This query centers on doing it. This query will help improve the process, because, when a problem is detected, the managers will have immediate information about who are the actors assigned to the business functions of the activity that is having problems and can correct it.

  Similar to the other queries, the output is given in a list format as seen in figure 5.15. Figure 5.16 shows the output for the activity "Return Car" of the *Rent-a-Car* process. As the other queries output, it begins with the business process's name and ID. Then, there is the searched activity's name and ID followed by the information of where to find it (i.e., Pool, lane, and, if it have, sub-lanes). Lastly, it references the actors, and the business functions of the activity that each actor is able to execute. In the case of the 5.16, only there are only two actors (”Beatriz Rodrigues” and ”Francisco Campaniço”) and each of them can execute all of the activity’s business functions.

In figure 5.17, there are presented different cases of output given by the query when modeling the permissions of the business process in different ways. For each of the cases consider that every time a lane has a arrow pointing to ”business functions” is because the lane contains a permission that grants access to those business functions. Consider also that there are two actors, Francisco Campaniço and Beateriz Rodrigues; the former having the require permission for the lane Worker 1 and the latter having the require permission for the lane Manager, every time there is one. In the first case (left business process), the output states that only Francisco Campaniço has access to the business functions of the searched activity (i.e., Act1). In the second case (middle business process), there is a permission associated to the lane Manager, and because of the inheritance property, the actor Beatriz Rodrigues has also access to all the business functions of the searched activity. Lastly, in the third case, the output changes and the business function BF1 only appears on the actor Beatriz Rodrigues. This because the business function is in the Manager lane which is only accessible by this actor.

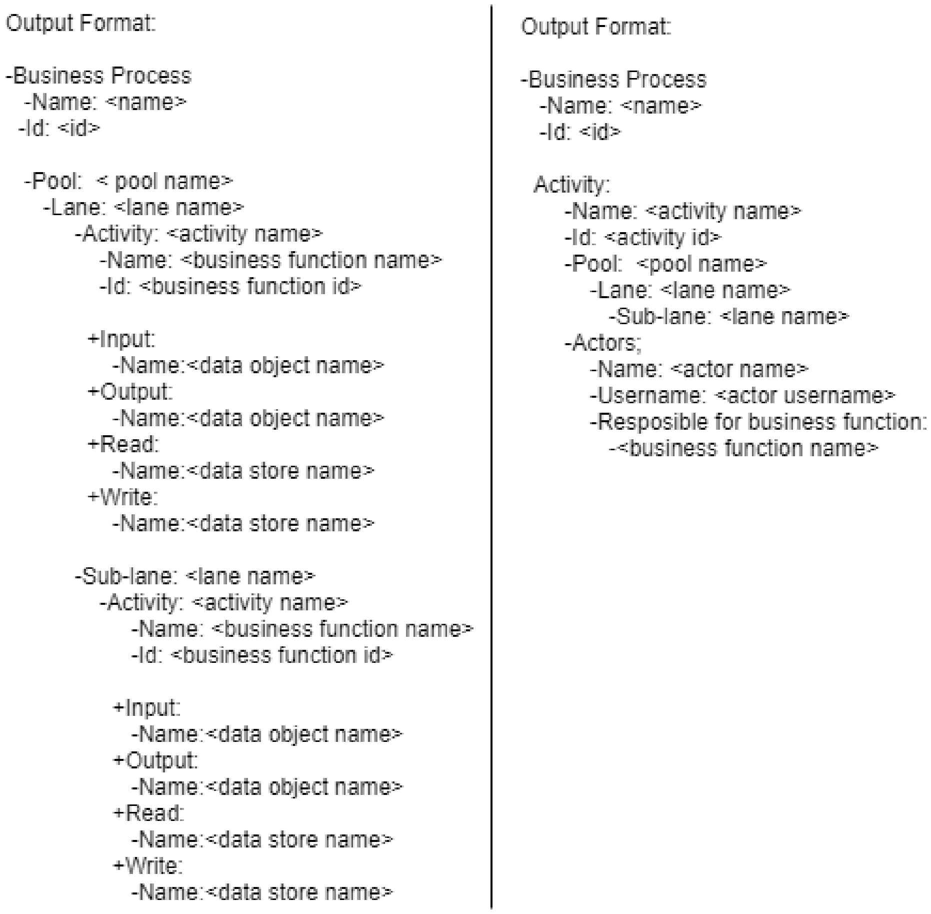


```
Output:

  -Pool:  Organization
     -Lane: Manager
          -Sub-lane: Worker 1
             -Activity: Act 1
                  -Name: BF1
                  -Id:
                  -Name: BF2
                  -Id:
                  -Name: BF3
                  -Id:
```

```
Output:

  -Pool:  Organization
     -Lane: Manager
          -Sub-lane: Worker 1
             -Activity: Act 1
                  -Name: BF1
                  -Id:
                  -Name: BF2
                  -Id:
                  -Name: BF3
                  -Id:
          -Sub-lane: Worker 2
             -Activity: Act 2
                  -Name: BF4
                  -Id:
                  -Name: BF5
                  -Id:
```

```
Output:

  -Pool:  Organization
     -Lane: Manager
          -Activity: Act 1
             -Name: BF1
             -Id:

          -Sub-lane: Worker 1
             -Activity: Act 1
                  -Name: BF2
                  -Id:
                  -Name: BF3
                  -Id:
          -Sub-lane: Worker 2
             -Activity: Act 2
                  -Name: BF4
                  -Id:
                  -Name: BF5
                  -Id:
```

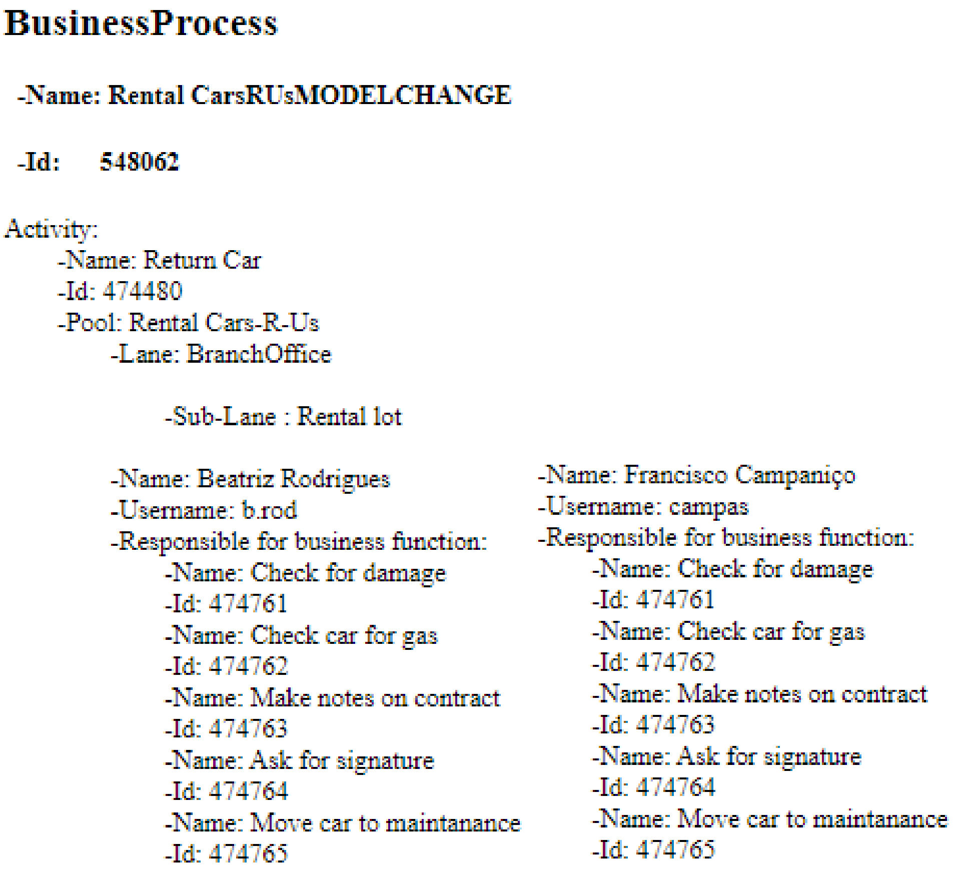
Figure 5.14: ”Show Business Process For Current User” query cases



```
Output Format:

-Business Process
   -Name: <name>
  -Id: <id>

   -Pool:  < pool name>
      -Lane: <lane name>
           -Activity: <activity name>
               -Name: <business function name>
               -Id: <business function id>

             +Input:
                -Name:<data object name>
             +Output:
                -Name:<data object name>
             +Read:
                -Name:<data store name>
             +Write:
                -Name:<data store name>

           -Sub-lane: <lane name>
               -Activity: <activity name>
                    -Name: <business function name>
                    -Id: <business function id>

                +Input:
                   -Name:<data object name>
                +Output:
                   -Name:<data object name>
                +Read:
                   -Name:<data store name>
                +Write:
                   -Name:<data store name>
```

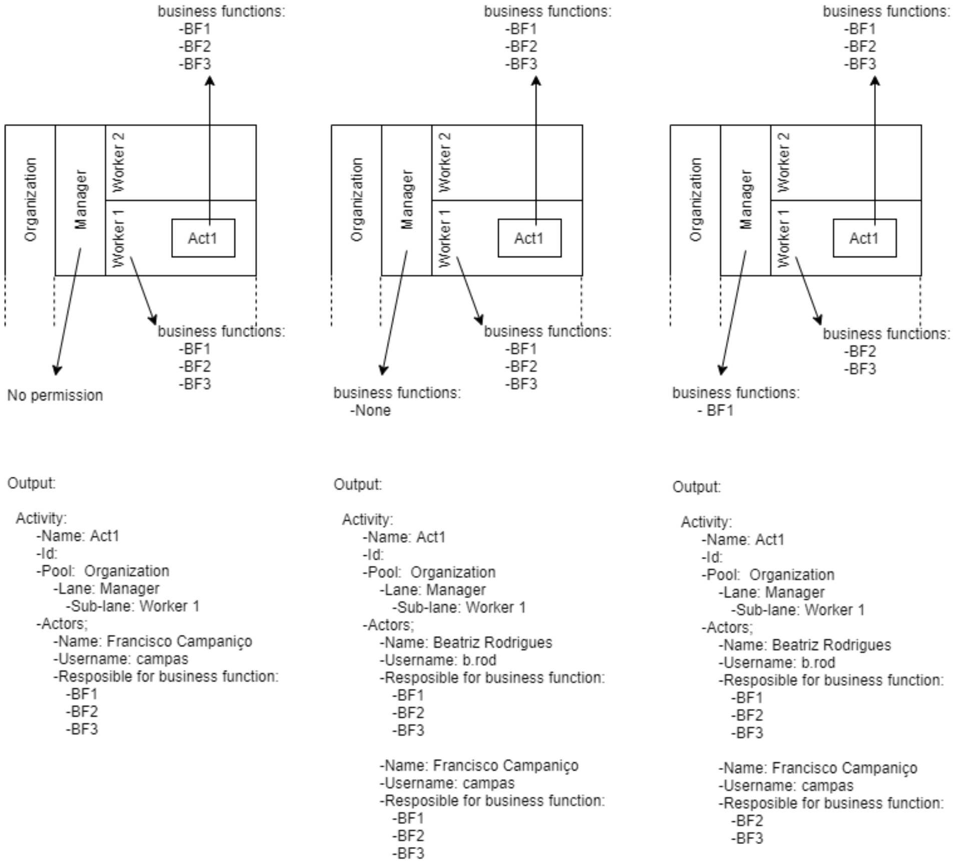
```
Output Format:

-Business Process
   -Name: <name>
   -Id: <id>

  Activity:
       -Name: <activity name>
       -Id: <activity id>
       -Pool:  <pool name>
           -Lane: <lane name>
              -Sub-lane: <lane name>
       -Actors;
           -Name: <actor name>
           -Username: <actor username>
           -Resposible for business function:
              -<business function name>
```

Figure 5.15: Output format of the "Show Business Process For Current User" (left) and "Show Actors of Activities" (right) queries



```
BusinessProcess

 -Name: Rental CarsRUsMODELCHANGE

 -Id:    548062

Activity:
    -Name: Return Car
    -Id: 474480
    -Pool: Rental Cars-R-Us
        -Lane: BranchOffice

            -Sub-Lane : Rental lot

        -Name: Beatriz Rodrigues
        -Username: b.rod
        -Responsible for business function:
            -Name: Check for damage
            -Id: 474761
            -Name: Check car for gas
            -Id: 474762
            -Name: Make notes on contract
            -Id: 474763
            -Name: Ask for signature
            -Id: 474764
            -Name: Move car to maintanance
            -Id: 474765

        -Name: Francisco Campaniço
        -Username: campas
        -Responsible for business function:
            -Name: Check for damage
            -Id: 474761
            -Name: Check car for gas
            -Id: 474762
            -Name: Make notes on contract
            -Id: 474763
            -Name: Ask for signature
            -Id: 474764
            -Name: Move car to maintanance
            -Id: 474765
```

Figure 5.16: Query's result for "Return Car" activity in *Rent-a-Car* business process



```
Output:

 Activity:
     -Name: Act1
     -Id:
     -Pool:  Organization
         -Lane: Manager
            -Sub-lane: Worker 1
     -Actors;
         -Name: Francisco Campaniço
         -Username: campas
         -Resposible for business function:
            -BF1
            -BF2
            -BF3
```

```
Output:

 Activity:
     -Name: Act1
     -Id:
     -Pool:  Organization
         -Lane: Manager
            -Sub-lane: Worker 1
     -Actors;
         -Name: Beatriz Rodrigues
         -Username: b.rod
         -Resposible for business function:
            -BF1
            -BF2
            -BF3

         -Name: Francisco Campaniço
         -Username: campas
         -Resposible for business function:
            -BF1
            -BF2
            -BF3
```

```
Output:

 Activity:
     -Name: Act1
     -Id:
     -Pool:  Organization
         -Lane: Manager
            -Sub-lane: Worker 1
     -Actors;
         -Name: Beatriz Rodrigues
         -Username: b.rod
         -Resposible for business function:
            -BF1
            -BF2
            -BF3

         -Name: Francisco Campaniço
         -Username: campas
         -Resposible for business function:
            -BF2
            -BF3
```

Figure 5.17: ”Show Actors of Activities” query cases

6

Conclusion

Contents

Today, business process models are the core elements of an organization and refer to how an organization is coordinated and how its work is organized to produce valuable products or services. When modeling business processes, policies that define the resources that can access and execute tasks or, informally speaking, with ' who can do what', are often not expressed. To solve this problem, we consider access control models as a way to offer the guarantee that only qualified users can gain access to tasks to execute them. To contribute with a solution for this problem, we propose the enforcement of an authorization approach based on access control models: integrate ABAC and RBAC with the specification of the BPMN, this way creating an extension of the BPMN standard. This will allow the specification of organizations' policies, related to resources, within the organizations' business process models, and prevention of the non-compliance behaviors by the actors involved in the tasks of the business processes.

The solution was implemented using Atlas, a project from Link Consulting. To do it, the related work of these topics was presented, and some of the concepts presented in them were followed to produce the solution. Most of the solutions available for integrating access controls are limited to the use of RBAC, a model that limits policies in terms of expressiveness, and associate roles to the tasks that actors can execute. This work proposes the use of ABAC for better representation of complex policies and allows also the use of RBAC for simple policies that involve only roles. Regarding business process compliance, two techniques exist *Forward-Checking Compliance*, and *Backward-Checking Compliance*. This work uses *Forward-Checking Compliance*, for checking that all activities in the business process model defined can be executable by at least somebody and proposes a system based on queries with the intention of simplifying the business processes for the actors, and for preventing non-compliance during the run-time of the process.

Thus, this work contributes by proposing a solution for representing resource policies in business processes by enforcing access control models, and integrating them into the BPMN meta-model. Then, implementing a framework for modeling the proposed extension of BPMN, and developing tool for checking processes and easing the execution of the process's instances for the actors involved.

6.1 Limitations and Future Work

This work has some limitations concerning other security requirement aspects. These are regarding the *Separation of duties*, and *Binding of duties* terms. The solution was not made thinking about these concepts. However, we think that these topics need to be thought and implemented to allow more options when modeling organizational policies. A way o implement the *Separation of duties* concept could be, to add a new attribute in the Lane class, where it is indicated the activities that need to be executed by different people, and in the designed tool have the activities split by different users with the lane's permissions. In the case of the *Binding of duties*, another attribute could be added to the Lane class that indicated the activities needed to be performed by the same user, and in the tool have assigned all the activities to only one user. Another limitation of the work is the representation of policies based on time, i.e., if a policy was for instance:

- ”The car preparation team can only prepare cars at Depot from 9 am to 5 pm”.

The part related to time could not be represented, since any of the access control models implemented are based on time or have a time component. For instance, this could be implemented by adding a time attribute to ABAC and cross it with the present date and hours to limit access.

Overall we believe that this line of work worth pursuing, and that what was develop in this work can be taken further in terms of the design of new control models to express better policies in BPMN, and in terms of more solutions for business process compliance. For future work, in addition to implementing the previous limitations, more specific queries should be develop to make the process even easier to the actors. Some of these should be regarding the sequence of the activities. For instance, giving a name of an activity and letting the actor know the next activity (as to the given as input) that they can execute is a helpful way for the actor to execute the process in more little steps. Another query helpful in the run-time of the process is related to the data elements, in specific data objects that correspond to documents. Consider the situation where an actor needs a document, produced in another lane, to execute a given activity. A query that given the name of the document, returns the activity where the document is produced, and who are the actors that can execute that activity will be helpful for the actor in need to know whom to contact to get the document. All these queries (implemented in the work and the proposed in this section) could be implemented inside Atlas to allow a visual representation too. This way the actors would have a simplified version of the process with the parts that they are allow to execute with a visual representation.

Lastly, this new information added to the business process about the access controls can be further use to restrict accessibility of users to applications. For instance, consider a business process that uses data stores. By using the extension of the language and seeing the activities where the data store is used, the code of the query ”Show relevant business process” can be easily adaptable to restrict the accesses to the database represented by the data store. This can not only be done with databases but also with other applications that a business process might use.

Bibliography

[1] S. L. Guerreiro, “On ontology of integration between access control and business process deep structure,” in *Novel Approaches to Information Systems Design*. IGI Global, 2020, pp. 226–246.

[2] M. Dumas, M. La Rosa, J. Mendling, and H. A. Reijers, *Fundamentals of business process management*. Springer, 2017, vol. 1.

[3] S. Guerreiro, “Conceptualizing on dynamically stable business processes operation: a literature review on existing concepts,” *Business Process Management Journal*, 2020. [Online]. Available: https://doi.org/10.1108/BPMJ-02-2020-0072

[4] C. Wolter and A. Schaad, “Modeling of task-based authorization constraints in bpmn,” in *International Conference on Business Process Management*. Springer, 2007, pp. 64–79.

[5] A. Elgammal, S. Sebahi, O. Turetken, M.-S. Hacid, M. Papazoglou, and W. van den Heuvel, “Business process compliance management : an integrated proactive approach,” in *Proceedings of the 24th International Business Information Management Association Conference, 6-7 November 2014, Milan, Italy*, K. Soliman, Ed. International Business Information Management Association (IBIMA), 2014, pp. 764–781, conference; 24th International Business Information Management Association Conference; 2014-11-06; 2014-11-07 ; Conference date: 06-11-2014 Through 07-11-2014.

[6] I. E. da Silva, “Conformidade dos processos de negócio: aplicação à modelação e operação de um processo de desenvolvimento de software,” Master’s thesis, Instituto Superior Técnico, 2019.

[7] D. R. Kuhn, E. J. Coyne, and T. R. Weil, “Adding attributes to role-based access control,” *Computer*, vol. 43, no. 6, pp. 79–81, 2010.

[8] OMG, *Business Process Model and Notation (BPMN), Version 2.0.2*, Object Management Group, Dec. 2013. [Online]. Available: http://www.omg.org/spec/BPMN/2.0.2

[9] R. S. Aguilar-Saven, “Business process modelling: Review and framework,” *International Journal of production economics*, vol. 90, no. 2, pp. 129–149, 2004.

[10] S. Guerreiro, “Enterprise dynamic systems control enforcement of run-time business transactions using demo: principles of design and implementation,” *Instituto Superior Técnico-Universidade Técnica de Lisboa, Lisboa*, 2012.

[11] K. C. Laudon and J. P. Laudon, *Management information systems*. Prentice Hall PTR, 1999.

[12] D. Gollmann, “Computer security,” *Wiley Interdisciplinary Reviews: Computational Statistics*, vol. 2, no. 5, pp. 544–554, 2010.

[13] D. Chadwick, A. Otenko, and E. Ball, “Role-based access control with x.509 attribute certificates,” *IEEE Internet Computing*, vol. 7, no. 2, pp. 62–69, 2003.

[14] B. Fisher, N. Brickman, P. Burden, S. Jha, B. Johnson, A. Keller, T. Kolovos, S. Umarji, and S. Weeks, “Attribute based access control,” *NIST Special publication*, p. 3B, 1800.

[15] A. Gordon and S. Hernandez, *The Official (ISC)2 Guide to the SSCP CBK*. John Wiley & Sons, 2016.

[16] M. Kurz, “Bpmn model interchange: The quest for interoperability,” in *Proceedings of the 8th International Conference on Subject-Oriented Business Process Management*, ser. S-BPM ’16. New York, NY, USA: Association for Computing Machinery, 2016. [Online]. Available: https://doi.org/10.1145/2882879.2882886

[17] M. El Kharbili, S. Stein, I. Markovic, and E. Pulvermüller, “Towards a framework for semantic business process compliance management,” *Proceedings of GRCIS*, vol. 2008, 2008.

[18] M. Fellmann and A. Zasada, “State-of-the-art of business process compliance approaches,” in *Proceedings of the 22nd European Conference on Information Systems (ECIS)*, 2014.

[19] C. Cabanillas, M. Resinas, and A. Ruiz-Cortés, “Hints on how to face business process compliance,” *III Taller De Procesos De Negocio E Ingeniería De Servicios, PNIS2010, Valencia, España*, 01 2010.

[20] M. e. Kharbili, A. K. A. d. Medeiros, S. Stein, and W. M. P. v. d. Aalst, “Business process compliance checking: Current state and future challenges,” in *Modellierung betrieblicher Informationssysteme (MobIS 2008)*, P. Loos, M. Nüttgens, K. Turowski, and D. Werth, Eds. Bonn: Gesellschaft für Informatik e.V., 2008, pp. 107–113.

[21] A. D. Brucker, I. Hang, G. Lückemeyer, and R. Ruparel, “Securebpmn: Modeling and enforcing access control requirements in business processes,” in *Proceedings of the 17th ACM Symposium on Access Control Models and Technologies*, ser. SACMAT ’12. New

York, NY, USA: Association for Computing Machinery, 2012, p. 123–126. [Online]. Available: https://doi.org/10.1145/2295136.2295160

[22] D. Nguyen, J. Park, and R. Sandhu, “A provenance-based access control model for dynamic separation of duties,” in *2013 Eleventh Annual Conference on Privacy, Security and Trust*, 2013, pp. 247–256.

[23] D. Basin, S. J. Burri, and G. Karjoth, “Separation of duties as a service,” in *Proceedings of the 6th ACM Symposium on Information, Computer and Communications Security*, 2011, pp. 423–429.

[24] K. Tan, J. Crampton, and C. A. Gunter, “The consistency of task-based authorization constraints in workflow,” in *Proceedings. 17th IEEE Computer Security Foundations Workshop, 2004.*, 2004, pp. 155–169.

[25] R. S. Sandhu and P. Samarati, “Access control: principle and practice,” *IEEE communications magazine*, vol. 32, no. 9, pp. 40–48, 1994.

[26] A. Rodriguez, E. Fernández-Medina, and M. Piattini, “A bpmn extension for the modeling of security requirements in business processes,” *IEICE Transactions on Information and Systems*, vol. E90D, 03 2007.

[27] A. Calabró, S. Daoudagh, and E. Marchetti, “Integrating access control and business process for gdpr compliance: A preliminary study.” in *ITASEC*, 2019.

[28] H. Klarl, C. Wolff, and C. Emig, “Identity management in business process modelling: A model-driven approach,” in *9. Internationale Tagung Wirtschaftsinformatik (WI2009), Wien, Österreich, 25.-27. Februar 2009*, 2009.

[29] E. F. Silva, D. C. Muchaluat-Saade, and N. C. Fernandes, “Across: A generic framework for attribute-based access control with distributed policies for virtual organizations,” *Future Generation Computer Systems*, vol. 78, pp. 1–17, 2018.

[30] E. Conrad, S. Misenar, and J. Feldman, “Chapter 6 - domain 5: Identity and access management (controlling access and managing identity),” in *CISSP Study Guide*, 3rd ed., E. Conrad, S. Misenar, and J. Feldman, Eds. Boston: Syngress, 2016, pp. 293 – 327. [Online]. Available: http://www.sciencedirect.com/science/article/pii/B9780128024379000060

[31] C. Wolter and C. Meinel, “An approach to capture authorisation requirements in business processes,” *Requirements engineering*, vol. 15, no. 4, pp. 359–373, 2010.

[32] W. Labda, N. Mehandjiev, and P. Sampaio, “Modeling of privacy-aware business processes in bpmn to protect personal data,” in *Proceedings of the 29th Annual ACM Symposium on Applied Computing*, 2014, pp. 1399–1405.

[33] K. S. Sang and B. Zhou, “Bpmn security extensions for healthcare process,” in *2015 IEEE International Conference on Computer and Information Technology; Ubiquitous Computing and Communications; Dependable, Autonomic and Secure Computing; Pervasive Intelligence and Computing*, 2015, pp. 2340–2345.

[34] M. E. A. Chergui and S. M. Benslimane, “A valid bpmn extension for supporting security requirements based on cyber security ontology,” in *International Conference on Model and Data Engineering*. Springer, 2018, pp. 219–232.

[35] V. C. Hu, D. Ferraiolo, R. Kuhn, A. R. Friedman, A. J. Lang, M. M. Cogdell, A. Schnitzer, K. Sandlin, R. Miller, K. Scarfone *et al.*, “Guide to attribute based access control (abac) definition and considerations (draft),” *NIST special publication*, vol. 800, no. 162, 2013.

[36] J. Dietz and H. Mulder, *Enterprise Ontology: A Human-Centric Approach to Understanding the Essence of Organisation*. Springer Nature, 01 2020.

[37] P. Harmon, *Business process change: a business process management guide for managers and process professionals*. Morgan Kaufmann, 2019.

A

Example of Atlas’s JSON File

Listing A.1: Part of Atlas' Rest API answer for request of information about Finance Lane of Rental Cars-R-Us business process

```
[
    {
        "name": "Name",
        "value": "Finance"
    },
    {
        "name": "BPMN Element",
        "value": [
            {
                "name": "Process Deposit Payment",
                "className": "BPMN Element",
                "label": "Process Deposit Payment",
                "id": 490336
            },
            {
                "name": "Check Deposit Payment",
                "className": "BPMN Element",
                "label": "Check Deposit Payment",
                "id": 477762
            },
            {
                "name": "Process Payment",
                "className": "BPMN Element",
                "label": "Process Payment",
                "id": 474474
            }
        ]
    },
    {
        "name": "Permissions",
        "value": [
            {
                "name": "PermissionFinanceABAC",
                "className": "PermissionABAC",
                "label": "PermissionFinanceABAC",
                "id": 497546
            }
        ]
    }
]
```

B

Sequence Diagram

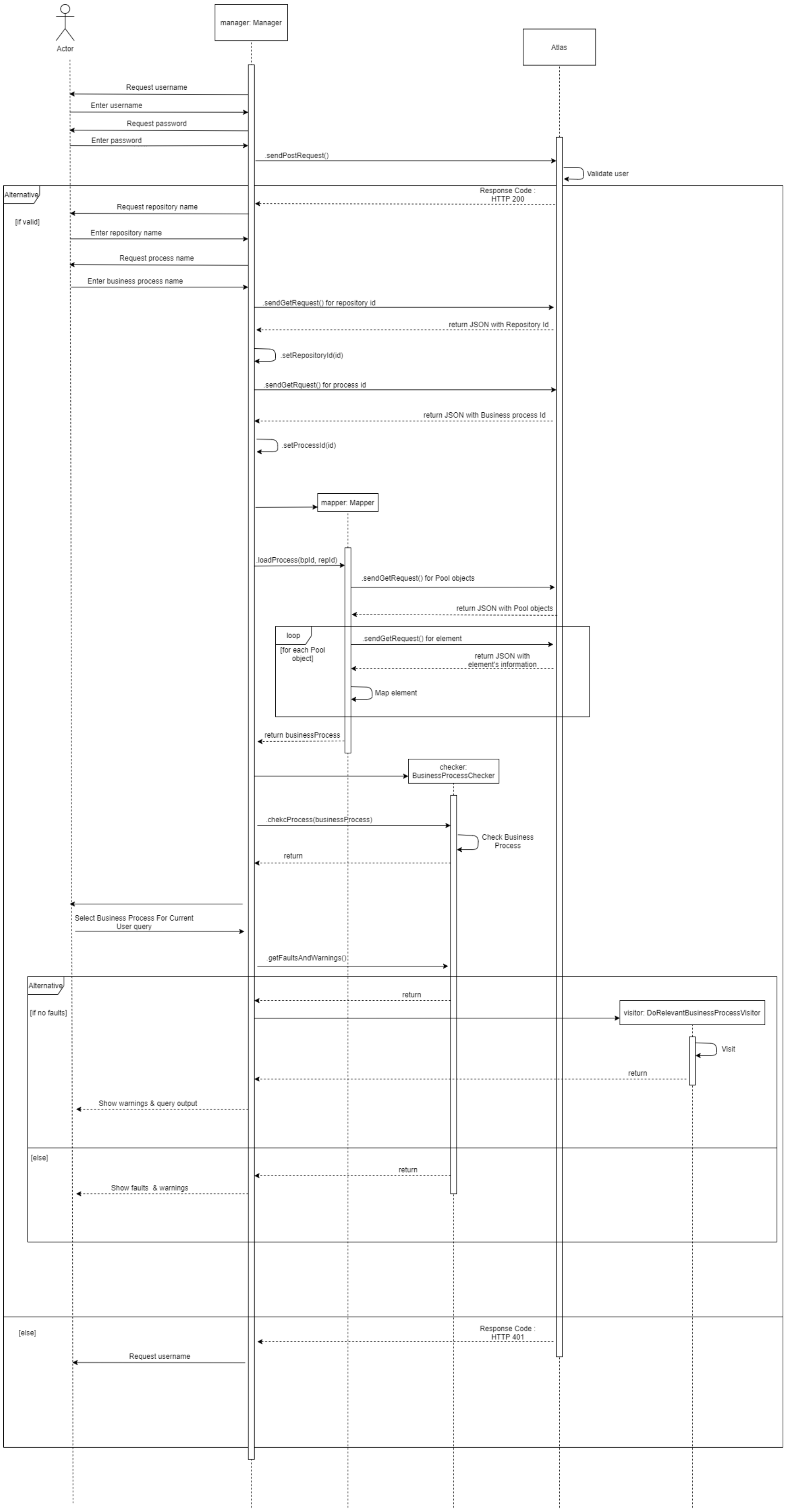


Figure B.1: Tool's sequence diagram simplified