ISCTE IUL
Instituto Universitário de Lisboa

Department of Information Science and Technology

Using a Test Automation Tool for Robotic Process Automation: an Empirical Study

Filipa da Silva Santos

Dissertation submitted as partial fulfillment of requirements for the degree of

Master in Computer Engineering

Supervisor:
Prof. Dr. Rúben Filipe de Sousa Pereira, Assistant Professor
ISCTE-IUL

July 2019

Acknowledgements

First, I would like to thank my supervisor, Professor Rúben Pereira for supporting me and always being available to give me suggestions to improve my research.

I would also like to thank my family and loved ones for encouraging me to persue my goals. Their advice and patience were fulcral throughout this journey. I will make up for all the weekends that I was working on this research instead of being with all of you.

Without their help, this would not have been possible.

Abstract

Robotic Process Automation (RPA) uses software robots that interact with systems through their user interface, reducing costs and improving efficiency in automating processes. Despite being a recent term, it is being progressively adopted in companies, being used in many areas, such as IT, Insurance and Human Resources.

Although RPA is relatively inexpensive to implement, the cost of licensing is high. To reduce costs, there are open source tools that might be capable of automating process, despite being used for other purposes, such as Test Automation. Therefore, this research aims to test if it is feasible to use a Test Automation tool to automate business processes and identify advantages and disadvantages of using a Test Automation tool as a RPA tool. To accomplish that, a Case Study (CS) was performed in a real company where the same business process was automated, using both a Test Automation tool and a RPA tool.

This research presents the comparison analysis and results of an experiment designed with two approaches: a Test Automation tool vs a RPA tool. The results show that despite there were some challenges of using a Test Automation to automate the selected process than with a RPA tool, using Test Automation tools may be useful for companies with low financial resources that aim to find low cost alternatives to RPA tools to automate processes, taking into account the advantages and disadvantages of using a Test Automation tool as a RPA tool provided in this research.

Keywords: Robotic Process Automation, RPA, Test Automation, Selenium

Resumo

Robotic Process Automation (RPA) utiliza *software robots* que interagem com os sistemas através da sua interface gráfica, reduzindo custos e melhorando a eficiência na automação de processos. Apesar de ser um termo recente, está a ser cada vez mais usado nas empresas, sendo usado em várias áreas, como o *IT*, seguradoras e recursos humanos.

Embora o *RPA* tenha um custo relativamente baixo de implementação, o custo das licenças é elevado. Para reduzir custos, há ferramentas *open source* que podem ser capazes de automatizar processos, apesar de serem usadas para outros fins, como *Test Automation.* Consequentemente, esta investigação pretende testar se é possível usar uma ferramenta de *Test Automation* para automação de processos e identificar vantagens e desvantagens de usar uma ferramenta de *Test Automation* como ferramenta de *RPA*.

Esta investigação apresenta a análise da comparação e resultados de uma experiência feita com duas abordagens: uma ferramenta de *Test Automation* e uma ferramenta de *RPA*. Os resultados mostram que apesar de terem sido identificados mais desafios ao usar uma ferramenta de *Test Automation* na automação do processo escolhido do que ao usar uma ferramenta de *RPA*, o uso de ferramentas de *Test Automation* pode ser útil em empresas com poucos recursos financeiros que procuram alternativas *low cost* às ferramentas de *RPA* para automação de processos, tendo em conta as vantagens e desvantagens do uso de uma ferramenta de *Test Automation* como ferramenta de *RPA* descritas nesta investigação.

Palavras-Chave: Robotic Process Automation, RPA, Test Automation, Selenium

Publications

This section presents the publications submitted in the context of this research, represented in Table 1.

Table 1 - Submitted publications

Journal	Rank	Decision
Business Process Management Journal	*Q1*	*Submitted*
Information and Software Technology	*Q2*	*Submitted*

Table of Contents

List of Tables

List of Figures

List of Abbreviations

AI	-	Artificial Intelligence
API	-	Application Programming Interface
APM	-	Application Portfolio Management
BP	-	Business Process
BPA	-	Business Process Automation
BPM	-	Business Process Management
BPMN	-	Business Process Model and Notation
BPO	-	Business Process Outsourcing
BU	-	Business Unit
CoE	-	Center of Excellence
CRM	-	Customer Relationship Management
CMDB	-	Configuration Management Database
CS	-	Case Study
ERP	-	Enterprise Resource Planning
FTE	-	Full-time Equivalent
GUI	-	Graphical User Interface
HTML	-	HyperText Markup Language
IDE	-	Integrated Development Environment
IS	-	Information Systems
IT	-	Information Technology
ITIL	-	Information Technology Infrastructure Library
OCR	-	Optical Character Recognition
PoC	-	Proof of Concept
ROI	-	Return on Investment
RPA	-	Robotic Process Automation
RQ	-	Research Question
STP	-	Straight Through Processing
UI	-	User Interface
WfM	-	Workflow Management Systems

Chapter 1 – Introduction

1.1. Background and Motivation

The Robots Revolution is on the rise. After the revolution that Customer Relationship Management (CRM) and Enterprise Resourcing Planning (EPR) created, a new term is going to revolutionize the workplace: Robotic Process Automation (RPA) (Anagnoste, 2017). This type of automation aims to automate business processes with the goal of improving efficiency while cutting costs (Cewe, Koch, & Mertens, 2017), by reducing the time humans spend dealing with Information Systems (IS), doing repetitive tasks such as typing, extracting, coping and moving huge amounts of data from one system to another system, meaning that these structured and manual tasks can be done by a robot, so that the workers can dedicate their time and effort to tasks that add more value (Aguirre & Rodriguez, 2017). Robots execute repetitive tasks by using Graphical User Interface (GUI) automation adaptors instead of Application Programming Interfaces (APIs) (as used in traditional automation) (Cewe et al., 2017), without changing the Information Technology (IT) infrastructure (Mindfields, 2015). This means that the robot does repetitive tasks that used to be done by humans faster and cost efficient.

Business units (BUs) spend a lot of time trying to improve business processes (BPs). Sometimes when IT cannot handle the requests from the BUs to modify an existing application or create custom applications, the BU must improvise (Slaby, 2012), by storing data in spreadsheets, for instance, which can have consequences, such as the data not being integrated into the used IT systems in the company, proneness to error and increase of security vulnerabilities. Using RPA, no expensive integration with systems is required (Mindfields, 2015) and the automation of BPs can be done by the BUs (M. Lacity, Willcocks, & Craig, 2016), without hiring offshore Full-time Equivalents (FTEs) (Slaby, 2012), with the advantage of being more familiar with the processes than the IT teams.

Despite RPA being a recent topic, being first used in 2012 and created by the marketing director Patric Geary (Sigurðardóttir, 2018), Forrester predicted that “by 2021, there will be more than 4 million robots doing office and administrative work as well as sales and related tasks” and that “RPA market will reach $2.9 billion by 2021 from $250 million in 2016” (Le Clair, 2018). These statistics show that the interest and adoption of RPA will continue to increase.

RPA can be used in multiple areas such as insurance (L. Willcocks, Lacity, & Craig, 2017), human resources (Hallikainen, Bekkhus, & Pan, 2018) and IT (Khramov, 2018) but not all processes are candidates for RPA automation. The most important features for automation are: frequent interaction with multiple systems, tasks that are prone to errors, ease of decomposition into unambiguous rules and voluminous transactions (Fung, 2014). Having these features is key to the success of RPA automation.

1.2. Problem and Research Questions

Despite RPA being relatively inexpensive to implement (Suri, Elia, & Hillegersberg, 2017) compared to a Business Process Management (BPM) solution, it is still quite expensive as the cost of RPA robots range from $5,000 to $10,000 annually (Le Clair, Cullen, & King, 2017). To reduce costs, there are open source tools, that despite being used for other purposes than RPA, such as test automation, it may be capable of automating processes, which can be useful for companies with low financial resources (Table 2).

Table 2 - Research Problem

Problem Id	Description
RP1	*RPA licensing cost is high (Le Clair et al., 2017).*

To reduce costs with licensing, a couple of research questions (RQs) were formulated to investigate if it is feasible to use a test automation tool to automate BPs and to find out the impact of it, identifying the advantages and disadvantages of using a test automation tool as a RPA tool (Table 3).

Table 3 - Research Questions

Research question Id	Description
RQ1	*Is it feasible to use a test automation tool as a RPA tool?*
RQ1.1	*What are the advantages and disadvantages of using a test automation tool as a RPA tool?*

To answer the RQs, the same process was automated with a test automation tool and with a RPA tool, using a case study methodology. Then, based on the findings, a comparison between the chosen RPA tool and the Test Automation tool was performed, in terms of their goal, implementation time, implementation difficulty, execution time, maintainability and technical aspects. Such comparison is then used to answer our RQs.

The novelty of this research lies on the attempt to automate the same business process with both approaches (a test automation tool and a RPA tool) since none research has investigated it before. This research intends to be a first step towards the exploration of this topic and does not provide an unequivocal answer to this problem, as there are many contexts where it can be explored. The results are useful for both professionals and academics.

Chapter 2 – Literature Review

Performing a Literature Review is adequate, because it can identify any gaps in the existing research to suggest areas for future research and can summarize the existing knowledge about a subject (Kitchenham, 2004). Therefore, this investigation follows the guidelines proposed on PRISMA Flow Diagram and also a concept-centric approach proposed by Webster and Watson (2002), as the review is guided by concepts, presented and discussed grounded on most relevant concepts related with RPA and Test Automation.

To elaborate this Literature Review, study searches about RPA and Test Automation were performed in major databases, such as Springer, IEEE, ACM, Google Scholar and ResearchGate between June and October 2018. The papers were collected based on their title, keywords and abstract. The results are represented in Table 4.

Table 4 - Number of papers collected by database and keyword

Database	Keyword	Number of papers collected
Springer	*Robotic Process Automation*	*7*
	Robotic Process Automation RPA	*0*
	Process Automation	*0*
	Test Automation	*2*
	Selenium	*1*
Google Scholar	*Robotic Process Automation*	*12*
	Robotic Process Automation RPA	*34*
	Process Automation	*0*
	Test Automation	*5*
	Selenium	*3*
IEEE	*Robotic Process Automation*	*2*
	Robotic Process Automation RPA	*3*
	Process Automation	*0*
	Test Automation	*1*
	Selenium	*5*
ACM	*Robotic Process Automation*	*2*
	Robotic Process Automation RPA	*0*
	Process Automation	*1*
	Test Automation	*0*
	Selenium	*3*
ResearchGate	*Robotic Process Automation*	*4*
	Robotic Process Automation RPA	*0*
	Process Automation	*0*
	Test Automation	*0*
	Selenium	*0*

Then, the findings were narrowed down according to the following criteria:

- Papers that address specifically RPA;
- Papers published in English;
- Papers electronically available on the internet.

To resume the search strategy applied, Figure 1 presents the process for paper selection. The search strategy performed is based on the PRISMA Flow Diagram (Moher, Liberati, Tetzlaff, & Altman, 2009).

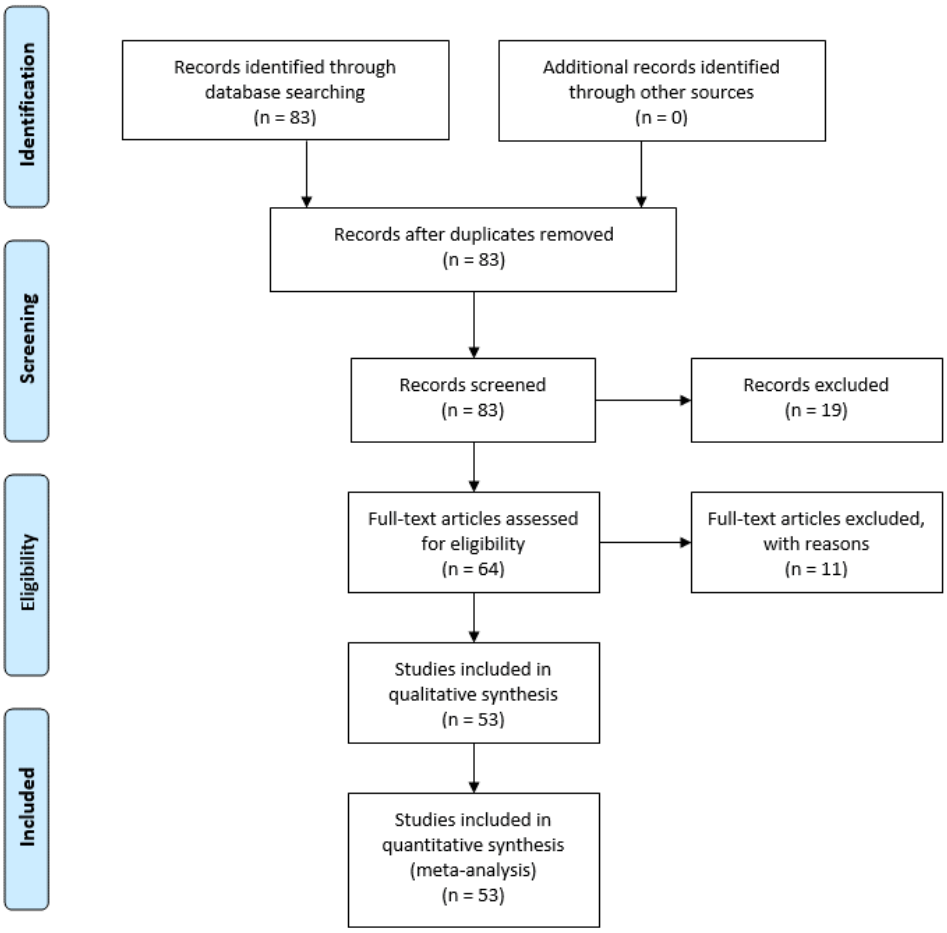

Identification
Screening
Eligibility
Included
Records identified through database searching (n = 83)
Additional records identified through other sources (n = 0)
Records after duplicates removed (n = 83)
Records screened (n = 83)
Records excluded (n = 19)
Full-text articles assessed for eligibility (n = 64)
Full-text articles excluded, with reasons (n = 11)
Studies included in qualitative synthesis (n = 53)
Studies included in quantitative synthesis (meta-analysis) (n = 53)

Figure 1 - Paper selection strategy

As far as the author could understand from the research performed, RPA research is a very recent topic and few researches exist, as can be seen in Figure 2. Nevertheless, one can see that the interest in RPA has been growing.

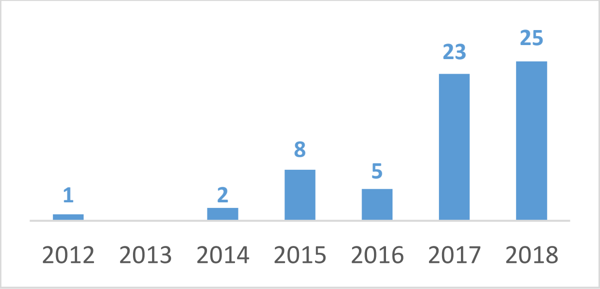


Figure 2 - Evolution of papers about RPA

Based on Figure 2, from 2017 on, the number of papers about RPA have grown rapidly. The tendency for the end of 2018 is to increase the number of RPA papers, as the number of papers from 2018 searched is from January to October.

In terms of Google searches, the number of searches about RPA has been increasing, as can be seen in Figure 3.

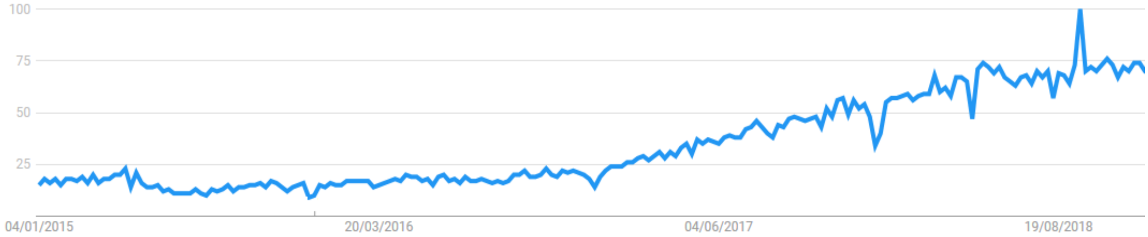


Figure 3 - Interest in RPA in terms of Google searches (retrieved from Google Trends)

From the end of 2016 on, the number of searches about RPA has rapidly increased and simultaneously the number of papers about RPA also increased, which demonstrates the increasing importance of RPA.

2.1. Robotic Process Automation (RPA)

2.1.1. RPA Overview

RPA is a recent term and has been increasingly adopted in companies. This term can be also referred as Service Automation (L. P. Willcocks, Lacity, & Craig, 2015) or Information Technology Process Automation (Fung, 2014). To understand what RPA means, it is necessary to define the term first.

IEEE has published a standard in 2017 about Intelligent Process Automation and defined RPA as “preconfigured software instance that uses business rules and predefined activity choreography to complete the autonomous execution of a combination of processes, activities, transactions, and tasks in one or more unrelated software systems to deliver a result or service with human exception management” (“IEEE Guide for Terms and Concepts in Intelligent Process Automation," 2017). This means that RPA is a software solution that executes repetitive tasks interacting with many systems, using business rules contained in processes.

This software solution uses robots to imitate human tasks (Geyer-Klingeberg, Nakladal, Baldauf, & Veit, 2018). These robots are not physical robots but software robots, meaning that a robot equals one software license (M. Lacity et al., 2016). To imitate human tasks, the robot uses GUI automation adaptors (Cewe et al., 2017) to interact with website user interface, like a human would do.

This type of automation is suited for "swivel chair" processes, where workers take inputs from many systems, process them and then add the processed inputs to other systems, like ERP and CRM (M. Lacity et al., 2016). Among the most performed tasks by robots are filling forms, logging into many systems, monitoring events, performing checks, sending emails and extracting data from many file types (Anagnoste, 2017).

To automate processes, there are many RPA tools, such as UiPath, Automation Anywhere and Blue Prism (Anagnoste, 2017). These vendors offer different features and some of them can be connected to other tools to add more functionalities from process mining, machine learning and artificial intelligence (Tornbohm & Dunie, 2017).

One definition of RPA points out that it is a kind of BPM (Cewe et al., 2017). This suggests that RPA may have its roots in BPM. From the research made, there were many terms that influenced RPA, represented in Table 5.

Table 5 - RPA Origin

Origin	**References**	**Number of references**
Business Process Automation (BPA)	*(Juntunen, 2018; Kämäräinen, 2018)*	*2*
Workflow automation	*(Aalst, Bichler, & Heinzl, 2018)*	*1*
BPM	*(Cewe et al., 2017)*	*1*
Screen Scraping	*(M. Lacity & Willcocks, 2015)*	*1*

One of the terms that influenced RPA is Workflow Automation (Aalst et al., 2018). In the nineties, Straight Through Processing (STP) was a hyped term. STP is adequate to processes that do not require human intervention, being a feature of Workflow Management Systems (WfM). (Aalst et al., 2018). WfM evolved into BPM, which is defined as the "concepts, methods, and techniques to support the design, administration, configuration, enactment, and analysis of business processes" (Weske, 2012). This concept involves many other concepts, such as reengineering, design and automation of processes. After these trends, BPA emerged, being defined as the automation of BPs, connecting the design of processes to application integration services, allowing workflow execution (Melchert & Winter, 2004). An important aspect is to distinguish BPA from BPM. BPM needs to define, re-define or optimize all processes on that domain before automating them, while BPA focuses on defining and analyzing a process when there is the need to be automated. In (Juntunen, 2018; Kämäräinen, 2018), RPA is referred as a subfield of BPA, which makes sense, since BPA and RPA both refer to automation of processes.

Finally, Screen Scraping is one of the factors that influenced RPA. Screen scraping captures mouse clicks and keystrokes on the user interface, relying on X and Y coordinates, but if the field was moved on the screen, this technology would not work (M. Lacity & Willcocks, 2015). On the other hand, RPA finds elements on the screen through HTML, Java Access Bridge and surface automation for Citrix (M. Lacity & Willcocks, 2015).

To summarize, RPA has roots in many terms, such as BPA (Juntunen, 2018; Kämäräinen, 2018), BPM (Cewe et al., 2017), Workflow Automation (Aalst et al., 2018) and Screen Scraping (M. Lacity & Willcocks, 2015). Some of these terms are related, meaning that Workflow Automation evolved into BPM (Weske, 2012) and later to BPA (Melchert & Winter, 2004).

2.1.2. Differences between BPM and RPA

Despite RPA being a kind of BPM (Cewe et al., 2017), it is important to distinguish BPM from RPA. RPA does not replace BPM, but complements it (M. Lacity et al., 2016), as each is suited to automate a certain type of business process. Table 6 presents the differences between BPM and RPA according to the literature.

The business goal of BPM is to reengineer processes (Forrester, 2014; M. Lacity et al., 2016). Once the process is reengineered, it is necessary to create a new application, as BPM interacts with other applications using APIs and interacting with the business logic and data access layers. RPA, on the other hand, has the goal to automate existing processes (M. Lacity et al., 2016), processes that are already defined and performed by humans, using a robot to replace them. RPA integrates with systems through the user interface, not requiring the creation of a new application to integrate with these systems, with the advantage of not requiring expensive integrations. By interacting with the user interface through the presentation layer, it does not change the logic of underlying systems and does not store transactional data (M. Lacity & Willcocks, 2015; M. Lacity et al., 2016).

Figure 4 illustrates the difference between the interaction of RPA and BPM with the systems.

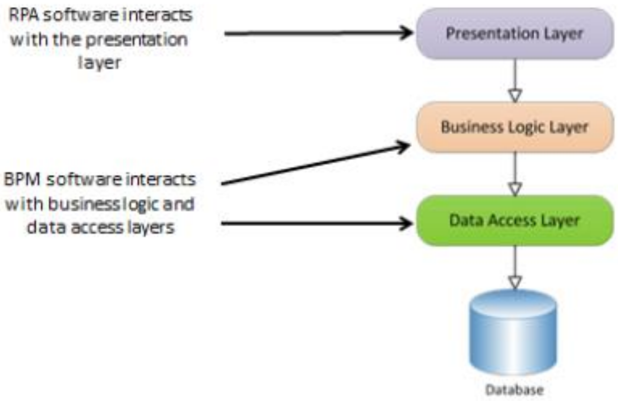

RPA software interacts with the presentation layer
BPM software interacts with business logic and data access layers
Presentation Layer
Business Logic Layer
Data Access Layer
Database

Figure 4 - Interactions of RPA and BPM with layers (M. Lacity et al., 2016)

As RPA interacts via front-end, it can be described as lightweight IT, whereas BPM can be described as heavyweight IT. Heavyweight IT can be described as back-end software that is IT owned, whereas lightweight IT can be described as front-end software owned outside IT (Sigurðardóttir, 2018).

BPM development may require integration with IT systems, such as ERP and CRM (Mindfields, 2015). As BPM integrates with these systems, it takes a long time to implement, therefore being best suited for processes that require expertise from IT and investments with high value (Suri et al., 2017). As BPM is IT owned, the automation development is done by programmers, requiring programming skills (Cewe et al., 2017; Khramov, 2018; M. Lacity & Willcocks, 2015). On the other hand, RPA is business owned and the development is made by the BU (M. Lacity et al., 2016), being suitable for processes that require business and process expertise (M. Lacity et al., 2016; Suri et al., 2017). Despite not having as much programming skills as programmers, it is not necessary as RPA software uses drag and drop (M. Lacity & Willcocks, 2015). Because all interactions with the applications are done through the user interface (Cewe et al., 2017), no complex integration is required and no change in IT infrastructure is needed, with the advantage of having a lower cost and fast development times (Mindfields, 2015).

Table 6 - Differences between BPM and RPA

Domain	BPM	RPA
Business goal	*Process reengineering (Forrester, 2014; M. Lacity et al., 2016)*	*Automation of existing processes (M. Lacity et al., 2016)*
Application	*Creation of a new application (Forrester, 2014; Khramov, 2018; M. Lacity & Willcocks, 2015; M. Lacity et al., 2016)*	*Use of existing applications (Aguirre & Rodriguez, 2017; M. Lacity et al., 2016)*
Integration Method	*Interacts with business logic and data access layers (Khramov, 2018; M. Lacity & Willcocks, 2015; M. Lacity et al., 2016)*	*Interacts with systems through the presentation layer (Aguirre & Rodriguez, 2017; M. Lacity & Willcocks, 2015; M. Lacity et al., 2016)*
Process Suitability	*Best suited for processes requiring IT expertise on high-valued IT investments (Suri et al., 2017)*	*Suitable for processes that require business and process expertise (M. Lacity et al., 2016; Suri et al., 2017)*
Programming Requirements	*Requires programming skills (Cewe et al., 2017; Khramov, 2018; M. Lacity & Willcocks, 2015)*	*Does not require programming skills (Aguirre & Rodriguez, 2017; Khramov, 2018; M. Lacity et al., 2016)*
Development Responsibility	*Development by programmers (M. Lacity et al., 2016; Suri et al., 2017)*	*Development by the business unit (M. Lacity et al., 2016)*
Deployment Times	*Long deployment times (Mindfields, 2015)*	*Fast deployment times – no complex integration required (Mindfields, 2015)*

To summarize the process suitability for both BPM and RPA described above, Figure 5 illustrates the relation between skills and technology investment for BPM and RPA.

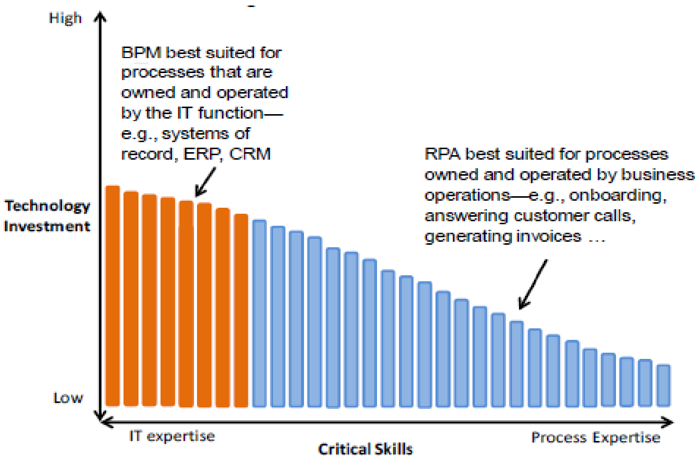


Figure 5 - RPA complements BPM (M. Lacity et al., 2016)

2.1.3. RPA Benefits

When implemented in a right manner, RPA can have multiple advantages. After analyzing the literature, the author has synthesized and summarized the main RPA benefits, described in Table 7. A more detailed and critical analysis of each elicited benefit is also presented in this section.

One of biggest advantages and the reason why companies are starting to use this technology massively is because of the fact that robots can work 24/7, replacing the work of 1.7 humans (Slaby, 2012), cutting entry costs to 70% (Anagnoste, 2017), therefore allowing FTE savings (M. Lacity & Willcocks, 2015; Suri et al., 2017; Tran & Ho Tran Minh, 2018). By replacing humans with robots to do repetitive work, allows workers to focus on more important tasks that involve problem solving and exception handling, improving job satisfaction and employee retention (Slaby, 2012). It can also create new jobs such as robot management, consulting and sophisticated data analytics (Asatiani & Penttinen, 2016) and reduce the dependence on offshore FTEs, using it to hire new FTEs (Slaby, 2012). To highlight this, an offshore FTE that costs $30000 can be replaced by a robot that can cost $15000 (Slaby, 2012).

Compared to humans, robots make less errors and work faster with more quality, therefore being more productive (Alberth & Mattern, 2017) and having a faster return on investment (ROI) (M. C. Lacity & Willcocks, 2017; Suri et al., 2017). This improves customer service, as customers are more satisfied with the job done by robots.

Table 7 - RPA Advantages

Advantages	**References**	**Number of references**
Can work 24/7 every day	*(Alberth & Mattern, 2017; Anagnoste, 2017; M. C. Lacity & Willcocks, 2017; M. Lacity & Willcocks, 2015; Slaby, 2012; Tran & Ho Tran Minh, 2018)*	*6*
Highly scalable/extensible/reusable solutions to meet peaks in service demand	*(M. C. Lacity & Willcocks, 2017; M. Lacity & Willcocks, 2015; Slaby, 2012; Suri et al., 2017; Tran & Ho Tran Minh, 2018; Vishnu, Agochiya, & Palkar, 2017)*	*6*
Performs tasks faster	*(M. C. Lacity & Willcocks, 2017; M. Lacity & Willcocks, 2015; Slaby, 2012; Suri et al., 2017; Vishnu et al., 2017)*	*5*
Less errors and consistent quality	*(Alberth & Mattern, 2017; M. Lacity & Willcocks, 2015; Suri et al., 2017; Tran & Ho Tran Minh, 2018)*	*4*
Allows employees to focus on more important tasks	*(M. Lacity & Willcocks, 2015; Slaby, 2012; Suri et al., 2017; Tran & Ho Tran Minh, 2018)*	*4*
FTE savings	*(M. Lacity & Willcocks, 2015; Suri et al., 2017; Tran & Ho Tran Minh, 2018)*	*3*
Deploys new functionalities faster than other IT solutions	*(Asatiani & Penttinen, 2016; M. Lacity & Willcocks, 2015)*	*2*
Integrates with systems through the application User Interface (UI)	*(Asatiani & Penttinen, 2016; M. Lacity et al., 2016)*	*2*
Fast ROI	*(M. C. Lacity & Willcocks, 2017; Suri et al., 2017)*	*2*
More productivity	*(Alberth & Mattern, 2017)*	*1*

As robots interact with the application user interface, they can integrate with every software, unconcerned of the openness to third party integration (Asatiani & Penttinen, 2016). Because the robot interacts with the user interface, the applications are not modified, having more security (Suri et al., 2017). This means that they can deploy new functionalities faster than other IT solutions that use APIs to integrate with systems, being implemented in 2-4 weeks, rather than months or years (Asatiani & Penttinen, 2016).

Robots can also adapt to service demand, being scalable, resizing fast without investing too much in development (Tran & Ho Tran Minh, 2018) and also can re-use components to help automating other tasks (Slaby, 2012).

2.1.4. RPA Disadvantages

Not only benefits are reported by the literature. It is also pointed some RPA disadvantages that companies must have into account when adopting RPA to automate processes. These disadvantages are synthesized and summarized in Table 8 and a more detailed analysis is presented further ahead.

One of the main disadvantages is that RPA is only suited for processes that are rule-based, because it is executed by a robot that lacks cognitive skills, needing rules in order to successfully execute its tasks. If the process contains a lot of exceptions, it must be handed to workers, increasing process complexity, as robot and human must be synchronized in order execute the tasks sequentially without any mistakes.

Table 8 - RPA Disadvantages

Disadvantages	**References**	**Number of references**
Only suitable for processes that include rule-based tasks	*(Alberth & Mattern, 2017; Asatiani & Penttinen, 2016)*	*2*
May be a temporary solution, which automates manual processes based on legacy IT systems	*(Asatiani & Penttinen, 2016)*	*1*
Increased process complexity when a part of the process still needs to be serviced by human workers	*(Alberth & Mattern, 2017)*	*1*
Creation of new tasks for the workers, as robots need to be supervised	*(Alberth & Mattern, 2017)*	*1*

As robots need to be supervised, there will be created new tasks for the workers to monitor the robot and guarantee that the outcomes of the execution are correct, which can reduce their time to execute tasks that add more value.

RPA can be a temporary solution to automate processes based on legacy systems, but in the long term it may be more appropriate to scrap the legacy system and build a new system (Asatiani & Penttinen, 2016).

2.1.5. RPA Suitable Processes Criteria

A critical information that may influence the success of the RPA implementation is the appropriateness of the process to be automated. It is important for organizations to know if a process is suitable for RPA implementation. Therefore, the author has looked in the literature for main criteria that a process must fulfill to be successfully automated

(Table 9). A more detailed description of each elicited criteria is also present in this section, so readers can better understand the boundaries of each criteria according to the literature.

Table 9 – RPA Criteria

Criteria	**References**	**Number of references**
Voluminous transactions	*(Asatiani & Penttinen, 2016; Fung, 2014; Kasslin, 2017; Kyheröinen, 2018; M. C. Lacity & Willcocks, 2017; M. Lacity, Willcocks, & Craig, 2015; Lintukangas, 2017; Slaby, 2012)*	*8*
Frequent interaction with multiple systems	*(Asatiani & Penttinen, 2016; Fung, 2014; Kasslin, 2017; Khramov, 2018; M. C. Lacity & Willcocks, 2017; M. Lacity et al., 2015; Slaby, 2012)*	*7*
Use of systems with a stable environment	*(Anagnoste, 2018; Asatiani & Penttinen, 2016; Fung, 2014; Kasslin, 2017; Khramov, 2018; M. Lacity et al., 2015; Slaby, 2012)*	*7*
Ease of decomposition into unambiguous rules	*(Asatiani & Penttinen, 2016; Fung, 2014; Khramov, 2018; M. Lacity et al., 2015; Slaby, 2012)*	*5*
No need or limited worker intervention	*(Fung, 2014; Lintukangas, 2017; Slaby, 2012; L. Willcocks et al., 2017)*	*4*
Limited need to handle exceptions	*(Asatiani & Penttinen, 2016; Fung, 2014; M. Lacity et al., 2015; Slaby, 2012)*	*4*
Awareness of current costs	*(Asatiani & Penttinen, 2016; Fung, 2014; M. Lacity et al., 2015; Slaby, 2012)*	*4*
Tasks prone to human errors	*(Asatiani & Penttinen, 2016; Fung, 2014; Khramov, 2018)*	*3*
High process maturity	*(Kyheröinen, 2018; Lintukangas, 2017; L. Willcocks et al., 2017)*	*3*
High level of process standardization	*(Kyheröinen, 2018; Lintukangas, 2017; L. Willcocks et al., 2017)*	*3*
High quality of data	*(Anagnoste, 2018; Lintukangas, 2017)*	*2*
Low need of cognitive requirements	*(Asatiani & Penttinen, 2016; Khramov, 2018)*	*2*
High availability of digital data	*(Anagnoste, 2018)*	*1*

First, it is important that the process can be decomposed into unambiguous rules, as RPA is only suited for rule-based tasks. Standardize the process before automating is also necessary because the more standardized the process is, the fewer exceptions happen (Lintukangas, 2017). Having not many exceptions to handle is a key factor, because having a lot of exceptions makes it time-consuming for the robot to automate (Slaby, 2012). Then, it is also important that the process is mature because a mature process can be easily measured, documented and stable, with a better current cost awareness.

Voluminous transactions are suitable for RPA automation because high volume (amount of repetition or time to complete the task) is considered as an opportunity for cost reduction (Lintukangas, 2017). Also, if the tasks are repeated often, it means that can be done by robots faster and with less errors.

Frequent interactions with multiple systems is also a good candidate for automation, as RPA interacts with systems through the presentation layer, whereas doing the same thing with traditional automation would be more expensive and time-consuming (Slaby, 2012). Another important feature is interacting with stable systems that do not change very often, so that the robot can interact with the interface without throwing exceptions that would be costly. A period of 12 to 18 months with no changes in the systems is excellent (Slaby, 2012).

Tasks that are prone to human error are suited for automation because it allows the reduction of costs and the increase of performance, as robots do less mistakes than humans. Also, tasks with no need or limited need for worker intervention and low cognitive requirements are an important aspect, because robots lack analytical and creative skills. Without the intervention of humans, the complexity of the process would increase.

Finally, data is important, in terms of digital availability and quality. To execute the tasks correctly, the data must be correct, so that the robot does not make mistakes and must be available digitally, to be accessible to the robot.

2.1.6. RPA Future Challenges

Another important aspect to have into account is the challenges that organizations may face during and after RPA implementation. Grounded on the literature, there are many challenges (Table 10) that should be addressed, so that in the future the adoption and implementation of RPA solutions can be more widespread. This section also presents a more detailed description of each challenge.

One of the main challenges is robot maintenance, as user interfaces change more often than the data structures behind it (Kasslin, 2017; Stople, Steinsund, Iden, & Bygstad, 2017). When systems change, sometimes the robot must be reconfigured, which is costly and time consuming.

Table 10 - RPA future Challenges

Challenges	**References**	**Number of references**
Robot maintenance	*(Kasslin, 2017; Stople et al., 2017)*	*2*
Competition between robots and humans	*(Asatiani & Penttinen, 2016; Suri et al., 2017)*	*2*
Can make mistakes faster	*(Kirchmer, 2017)*	*1*
Robot having wide access rights	*(Kasslin, 2017)*	*1*
Unclear division of responsibilities between IT and BUs	*(Suri et al., 2017)*	*1*
Lack of understanding of what RPA means and its application	*(Suri et al., 2017)*	*1*

There is also lack of understanding of what RPA means and its application, because the term itself is confusing as it suggests that is connected to robotics, but instead, it is related to software robots.

As there is no human checking before executing a task, the robot can make mistakes faster, not waiting for the responses from the applications, like a human would do and not being able to check connection problems, performing only a part of its tasks. Also, a robot can have wide access rights to interact with other systems, having as many accesses as a super user, which can arise security issues.

The responsibility of implementing RPA is on the business side, but sometimes there is an unclear division between this side and the IT side. This happens because RPA is an IT tool, but at the same time automates processes that belong to the business side.

The last challenge is the impact on employees. Many companies allocate their workers into other tasks, while others just replace their workers with robots. Despite the positive feedback without many job losses, employees are still reluctant and see robots as their opponents for a job, which creates tension on the workplace, so it is important to address these issues with workers before introducing robots into the workplace (Asatiani & Penttinen, 2016).

2.1.7. RPA Future Opportunities

It is not only interesting to explore the benefits, disadvantages, main criteria and challenges but also possible opportunities to improve the final results of RPA implementation (Table 11). The combination of different techniques/domains is actually seen as a suitable opportunity. Moreover, it may even solve some of the RPA disadvantages aforementioned (Table 8). There are many opportunities that, if well applied, can leverage the adoption of RPA to another level. This section also provides a more detailed description of each opportunity found in the literature.

The most obvious is the integration with machine learning and artificial intelligence. As RPA is only suited for rule-based processes, this integration can help handling with a higher range of BPs, overcoming this limitation. There are RPA vendors with Artificial Intelligence (AI) integration. Some of them incorporate their own algorithms in their software, such as WorkFusion and Pegasystems, while others (e.g. Blue Prism and Kryon) link to other platforms, such as IBM Watson or Microsoft Azure ML (Le Clair et al., 2017). There will also be an integration with text mining, enhancing the value of RPA by being able to handle with unstructured content and extract features like intentions and sentiments (Le Clair, 2018).

Table 11 – RPA Opportunities

Opportunities	**References**	**Number of references**
Integration with Machine Learning/AI	*(Aalst et al., 2018; Anagnoste, 2017; Khramov, 2018; Le Clair et al., 2017; Tornbohm & Dunie, 2017)*	*5*
Analytics will eventually come to RPA	*(Anagnoste, 2017; Le Clair, 2018; Le Clair et al., 2017)*	*3*
Integration with process mining	*(Aalst et al., 2018; Tornbohm & Dunie, 2017)*	*2*
Integration with text mining	*(Le Clair, 2018)*	*1*

Also, analytics with RPA can be advantageous, because robots will be able to use data and also interpret it like humans would do, detecting patterns faster and without making mistakes. By analyzing data, RPA combined with process mining (Aalst et al., 2018; Tornbohm & Dunie, 2017) can detect processes suited for automation. There are already collaborations between process mining and RPA vendors, such as UiPath and Celonis, to select processes and then build robots driven by the selected processes (Aalst et al., 2018).

2.2. Test Automation

2.2.1. Test Automation Overview

Test automation can be defined as the "use of testing tools to reduce the need of manual or human involvement, repetitive or redundant tasks" (Singla & Kaur, 2014). This means that by automating tests, they can be executed without the need of human involvement.

Finding defects (Raut, Chudiwale, & Kawale, 2016) is the goal of test automation. Test automation is used to compare the pretended outcome with the actual outcome, validating if a functionality works as intended (Jain & Kaluri, 2015). There are many types of tests, such as unit tests, integration tests, regression tests and GUI tests. GUI tests test the outcomes of UI events and check if they are correct (Patil & Temkar, 2017). In this research, a GUI test automation tool is used to automate a process.

After researching, test automation has many advantages, represented in Table 12.

Table 12 - Test Automation Advantages

Advantage	**References**	**Number of references**
Unlimited repetitions of test cases	*(Patil & Temkar, 2017; Vila, Novakova, & Todorova, 2017)*	*2*
Finding defects missed in manual testing	*(Leotta, Clerissi, Ricca, & Spadaro, 2013; Patil & Temkar, 2017)*	*2*
Efficiency - Execution time of automated tests is less than executing them manually	*(Leotta et al., 2013; Vila et al., 2017)*	*2*
Rapid feedback to developers	*(Patil & Temkar, 2017)*	*1*
Expenses decrease	*(Vila et al., 2017)*	*1*

One of the advantages is that tests can be executed every time (Patil & Temkar, 2017; Vila et al., 2017), eliminating the risk of a human making a mistake. It also helps finds defects missing in manual testing (Leotta et al., 2013; Patil & Temkar, 2017), saving a lot of time in testing and therefore increasing efficiency (Leotta et al., 2013; Vila et al., 2017).

By finding defects more easily, feedback is given rapidly to developers (Patil & Temkar, 2017), allowing them to solve bugs faster, decreasing expenses (Vila et al., 2017).

Despite its advantages, Test Automation also has some disadvantages, represented in Table 13.

As tests are built on top of web pages, these pages may change and therefore the test breaks, so the test has to be recorded again (Leotta et al., 2013; Patil & Temkar, 2017), which is a time consuming task and an expensive task. Sometimes it is not adequate to use automated tests, but instead it should be used manual testing, especially if the web page being tested changes its user interface very often.

Table 13 - Test Automation Disadvantages

Disadvantage	References	Number of references
When web pages are frequently updated, the test suite must be recorded again - time consuming task	*(Leotta et al., 2013; Patil & Temkar, 2017)*	*2*
Small changes in a web page (e.g., in the page layout) may break test cases	*(Leotta et al., 2013)*	*1*
Recording some activities in a wrong manner	*(Patil & Temkar, 2017)*	*1*

Also, it may record actions wrongly (Patil & Temkar, 2017) and then test case breaks, so developers have to record it again, which has an impact in costs and time.

2.2.2. Test Automation Tools Comparison

There are several test automation tools to test user interfaces. Table 14 shows a list of these tools.

Table 14 - Test Automation Tools

Tool	References
Selenium	*(Bajaj, 2017; Gupta, Kumar, & Saxena, 2015; Islam, 2016; Kaur & Gupta, 2013; Shaukat, 2015; Sualim, Yassin, & Mohamad, 2016)*
QTP/UTF	*(Bajaj, 2017; Gupta et al., 2015; Islam, 2016; Kaur & Gupta, 2013; Shaukat, 2015)*
Watir	*(Bajaj, 2017; Islam, 2016; Shaukat, 2015; Sualim et al., 2016)*
Test Complete	*(Kaur & Gupta, 2013; Shaukat, 2015; Sualim et al., 2016)*
Geb	*(Bajaj, 2017; Shaukat, 2015)*
FitNesse	*(Shaukat, 2015)*
Tricentis Tosca	*(Shaukat, 2015)*
Silk Test	*(Shaukat, 2015)*

In order to pick the test automation tool to be used in this research, the 3 most referenced tools in the literature were analyzed. Table 15 represents the comparison of test automation tools based on several features.

As can be seen, Selenium has many advantages in comparison with QTP/UTF, because it is open source, supports all platforms and browsers and supports more languages than QTP/UTF. On the other hand, Selenium only supports web applications, whereas QTP/UTF supports desktop and web applications, not requiring programming skills like Selenium does. Compared with Watir, Selenium supports more languages than

Watir and has support from the online and professional community, whereas Watir has only community support.

Table 15 - Test Automation Tools comparison (adapted from Shaukat (2015))

Feature	Selenium	QTP/UTF	Watir
License	*Open Source*	*Commercial*	*Open source*
Operating systems	*Cross Platform*	*Windows*	*Cross Platform*
Browser support	*All browsers*	*IE, Firefox, Chrome*	*All browsers*
Language Support	*Java, C#, PHP, Ruby, Python, Perl*	*VB Script*	*Ruby*
Application support	*Web*	*Desktop and web*	*Web*
Programming skills	*Yes*	*No*	*Yes*
Community support	*Online/professional community*	*HP support*	*Online community*

In summary, Selenium is the most cited test automation tool, being open source, supporting more languages than the other two referred tools and having an online and professional community and for those reasons, it is test automation tool used in this research.

2.2.3. Selenium

Selenium is a test automation tool used to test web-based applications (Patil & Temkar, 2017; Ramya, Sindhura, & Sagar, 2017). It interacts with web pages by locating the page elements, with many options for locating them, such as by id, name and XPath. By using JavaScript to embed the automation engine into the web browser, it allows to test scripts on many browsers (Holmes & Kellogg, 2006; Patil & Temkar, 2017).

Selenium has many tools, each one with a specific purpose to help with the development of scripts. They are Selenium IDE, Selenium Webdriver and Selenium Grid (Patil & Temkar, 2017):

- Selenium IDE - plugin for Firefox and Chrome that helps users to record, edit and debug tests. It can record user actions and then execute them automatically;
- Selenium WebDriver - testing tool that communicates with the browser through the browser driver, that sends commands to the browser and receives responses. It supports many languages to write scripts, such as Java, Python, C# and PHP. To automate tests, it locates the elements on the web page,

working well on dynamic pages where the elements change without reloading the page;

- Selenium Grid - used to run tests in multiple browsers at the same time, ensuring that tests are compatible with many versions and browsers, saving time in the execution of tests. It uses the hub and nodes concept, where the hub is used to run the test, but the execution is done on the nodes.

One author identified several advantages of Selenium (Vila et al., 2017). One of the advantages is the ability of supporting many browsers (Firefox, Chrome, Internet Explorer and others), mobile (Android and iOS), platforms (Windows, Mac OS and Linux), programming languages (Java, C#, Ruby, Python and others) and IDEs (Eclipse, IntelliJ IDEA, NetBeans and Visual Studio). As an open source tool, source code is available and can be modified by users and has also has an active community to solve issues. Another advantage is simulating a user working with a browser, being more accurate and faster to execute tests compared to a user. It also handles well dynamic content by using appropriate locators to locate elements.

Despite having many advantages, there are some disadvantages, described in Table 16.

Table 16 - Selenium Disadvantages

Disadvantage	**References**	**Number of references**
Needs programming skills	*(Vila et al., 2017)*	*1*
Supports only web applications	*(Altaf, Dar, Rashid, & Rafiq, 2015)*	*1*
Selenium sometimes cannot automate when: • *The application is a flash application* • *The application has custom objects or objects that cannot be inspected*	*(Altaf et al., 2015)*	*1*

One of the disadvantages is the need of programming skills. If someone wants to edit tests or add more functionalities, programming skills are required (Vila et al., 2017).

Selenium, as said before, supports only web applications (Altaf et al., 2015), being impossible to locate elements outside the scope of a web application. Even when the application being tested is web-based, Selenium cannot automate applications, such as flash applications or when the application has custom objects or objects that cannot be inspected (Altaf et al., 2015).

Chapter 3 – Related Work

There has been a lot of research about BPM throughout the years, whereas research on RPA is recent and scarce (Leopold, Aa, & Reijers, 2018).

To systematize the related work found, Table 17 lists the most relevant papers found by some features, such as the process automated, company name, country, industry and software tool. Along these researches, RPA is used in multiple industries, such as human resources, mobile communications, insurance and BPO, using UiPath and Blue Prism to automate processes. As can be seen, no papers reported the use of a test automation tool for RPA. This reinforce the novelty of our findings.

Table 17 - RPA Related work

Paper	Process	Company	Country	Year	Industry	Tool
(Khramov, 2018)	*Customer ticket processing*	*Basware Oy*	*Finland*	*2018*	*IT*	*UiPath*
(Aguirre & Rodriguez, 2017)	*Generation of a payment receipt*	*Not present*	*Colombia*	*2017*	*Business Process Outsourcing (BPO)*	*Not present*
(L. Willcocks et al., 2017)	*Premium advice notes processing*	*Xchanging*	*UK*	*2015*	*Insurance*	*Blue Prism*
(Hallikainen et al., 2018)	*Quality control of employee data*	*OpusCapita*	*Finland*	*2014*	*Human Resources*	*UiPath*
(M. Lacity et al., 2016)	*SIM swaps*	*Telefonica O2*	*UK*	*2010*	*Mobile Communications*	*Blue Prism*
(M. Lacity et al., 2016)	*Pre-calculated credit*	*Telefonica O2*	*UK*	*2010*	*Mobile Communications*	*Blue Prism*

The automated process in (Aguirre & Rodriguez, 2017) was a payment receipt generation. This process started when the client requests a payment receipt. The Front office agent receives the request and creates the case on the CRM system and then the back-office agent opens the case, generates the receipt and sends it by email to the client. With RPA, the back-office tasks are assumed by a robot. The process with RPA could deal with 21% more cases than with the process without RPA, improving productivity. Mean case duration was also evaluated. The process with RPA lasted just 9 seconds less than without RPA. This could be explained by the workers performing their activities as fast or faster than the robot, but the robot has the advantage of being able to deal with multiple cases simultaneously.

A Finnish technology company called Basware Oy used UiPath to automate the processing of customer tickets (Khramov, 2018). To improve the processing, they used RPA to classify and assign tickets belonging to the first line. The process started when incoming tickets were assigned to the first line. Then it detected the language and the English tickets were sent to a machine learning classifier, that predicts the appropriate group. After testing the automation, the speed of processing tickets by the robot was analyzed and compared to a human doing the same tasks. While the human took 2 minutes to complete the tasks, the robot took 10 seconds.

Another Finnish company called OpusCapita also used UiPath to automate processes related to human resources (Hallikainen et al., 2018). The chosen process was the quality control of employee data. The process started with the transfer of new and modified employee records to a payroll management system. Each night, the robot logs in this system and checks if there were any mistakes in the employee records and if there are, it corrects them. After that, the robot sends an email with the errors found and corrected. After the implementation, some problems occurred. One of them was not considering that sometimes the robot did not wait for the responses from the applications, working too fast. Adding timeouts solved the problem. The other problem was the internet connection was not always operational, causing the robot to perform part of the activities. Despite these problems, the company noted that the robot could perform the tasks faster and with less mistakes than humans.

Xchanging was another company that used RPA to automate processes, using Blue Prism (L. Willcocks et al., 2017). They had a lot of manual tasks that need to be automated. One of them was the creation of premium advice notes (LPAN), so that insurance brokers can submit premiums to Xchanging. Once the LPAN was created, it was uploaded to the insurer’s repository. Then the customer had to send an unstructured file that had to be opened and validated. When the process was automated, the robot had to validate the data sent, deciding if the request should be completed or make an exception. The exceptions were handled by humans. This automation reduced the time of LPAN processing. Before, processing 500 LPAN tooks many days and now the robot can do it in 30 minutes, without making mistakes.

Telefonica O2 conducted two pilots in 2010 using Blue Prism (M. Lacity et al., 2016). One of the processes was replacing the customer's SIM with a new one but keeping the person's number. The other one was the application of a pre-calculated credit to a client's account. These two pilots were beneficial to Telefonica O2, performing the tasks as expected. Despite its benefits, there were some negative ideas ideas by the IT team regarding RPA, so they wanted to test if RPA can get the same results as BPM. To test it, the IT team automated the two processes referred above. In terms of implementation time, the results were similar, taking three weeks to implement. Nonetheless, on the financial side, RPA won. Despite RPA having more costs with training and support, the development cost was lower, as RPA needs less IT labor than BPM. While for 10 automated processes BPM systems would take three years to payback, with RPA the same processes would take 10 months to payback.

These case studies demonstrate that RPA can be applied to many processes from different areas, such as IT, insurance, human resources and mobile communications. The benefits reported include more productivity, increased speed, dealing with many cases simultaneously, reduction of errors and faster ROI. However, there are some issues that must be considered, such as the robot working too fast, which can produce more mistakes if the speed of the robot is not considered, connection problems, which can make the robot not being able to complete the tasks ahead and also suspicion from employees regarding RPA, not understanding what RPA means and where can be applied. To successfully automate processes, it is necessary to communicate where RPA can be applied to employees and to find out if the process to be automated is suitable, establishing criteria to find the most suited processes. These companies applied these principles and learned from their mistakes, allowing them to successfully automate processes.

While the case studies described above focused on automating processes with RPA tools, such as UiPath and Blue Prism, this research will focus on finding out if it is possible to automate a process using a test automaton tool, such as Selenium, eliciting pros and cons comparing it to a RPA tool.

Chapter 4 – Research Methodology

The research methodology used in this research is Case Study (CS). This research method aims to explore a contemporary and real-life phenomenon, through the analysis of the relationship between a number of events (Zainal, 2007), in which the researcher has limited control or no control at all (Yin, 2009). It focuses on the particular rather than the general, being appropriate to gain insights from the information gathered from the particular focus (Thomas, 2011), by viewing its many angles.

As stated by Yin (2009), a CS is built around research questions, guiding the whole research. In this research, two research questions were defined in Table 3. One of the questions is a "what" question, which confirms the exploratory nature of this CS, as its goal is to establish propositions for further inquiry. A CS may also have "why" and "how" questions (Perry, Sim, & Easterbrook, 2004).

These questions set up for phenomenon exploration, being appropriate for studying a phenomenon where there is not former work (Yin, 2009; Zainal, 2007). As RPA is a recent concept and there are no studies that used Selenium to automate a business process, this methodology is appropriate, as it studies a phenomenon where there is lack of research about it (Yin, 2009), therefore this is an exploratory CS.

CSs are meant to use multiple sources of data to gather the many viewpoints of the participants (W. Tellis, 1997). This process, known as triangulation, ensures that the problem is explored through many lenses (Baxter & Jack, 2008), providing a better picture, which increases the precision of the conclusions (Runeson & Höst, 2009).

Thomas (2011) argues that CSs should be classified according to their subject, purpose, approach and process to organize the research. The first classification, the subject, is choosing a focus, whether it is an unusual (outlier case), familiar (local case) or a good example (key case). Besides deciding what is the research subject, it is important to define its purpose (explaining or evaluating or exploring a phenomenon), its approach (testing a theory, describing, interpreting or build a theory) and its process (how will the CS be done).

Figure 6 illustrates the different classifications of this CS, according to Thomas (2011).

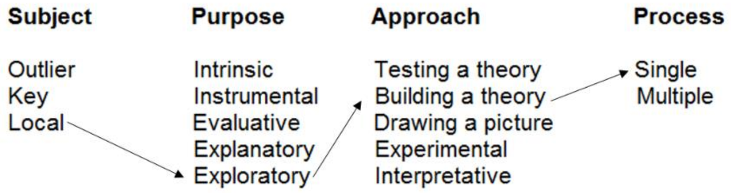


Figure 6 - Classifications of this CS

As illustrated in Figure 6, this CS is classified as a local case, since it is familiar to the researcher, as he works in the environment where the CS will take place.

In terms of the purpose, a CS can be intrinsic (undertaken because of its interest), instrumental (with a purpose in mind), evaluative (to check how well something is working), explanatory (seeks an explanation in the form of a causal relationship) and exploratory. As mentioned before, this CS is exploratory as it aims to explore a phenomenon where there is lack of research about it. Regarding the approaches defined by Thomas (2011), a CS can test a theory (when there is already research about the phenomenon), draw a picture (illustrates a phenomenon), experimental (test of ideas under certain conditions), interpretative (based on the literature) or build a theory (develop from scratch). According to the definitions provided, this CS focuses on building theory as there are no studies about the use of Selenium for RPA so far. The last classification is defining whether the CS is a single CS (focuses on studying an event limited by a single occurrence or a single unit of analysis) or a multiple CS (many studies studied together to investigate a phenomenon). This research is a single CS, because it focuses on a single team, which is the single unit of analysis, as stated by Yin (2009).

This CS will be divided in four stages, as defined by Yin (1994), represented in Table 18.

Table 18 - CS Stages (Yin, 1994)

Stage	**Stage Description**
Design the CS	*Definition of objectives and CS planning.*
Conduct the CS	*Data collection preparation, definition of protocols for data collection and data collection execution.*
Analyse CS evidences	*Analysis and evaluation of the data gathered.*
Develop conclusions	*Conclusions based on the data analysis.*

To design and conduct a CS, it is necessary to have the required skills, such as being a good listener, ask good questions, being unbiased, adaptative and flexible (Yin, 2009). Having a protocol is important, as it provides more reliability to the research (W. M.

Tellis, 1997). A protocol should include an overview of the CS in terms of objectives and issues, field procedures (credentials to access data, sources of data and procedure reminders), case study questions and a guide for the CS report.

After defining the protocol, this research describes the selected process, contextualizing it in terms of the activities, number of people that performed the process manually, cost per hour, periodicity, time to perform the process and each task, using multiple sources of data, such as semi-structured interviews, observation and process documentation. Then, an explanation on how each activity was implemented in Selenium and UiPath (a RPA tool) is presented, exposing difficulties and workarounds of the implementation. The execution time of each activity is also elicited, comparing and exposing the differences of times between both approaches. Plus, it compares both approaches in terms of difficulty of implementation, implementation time, among others, to extract the advantages and disadvantages of using Selenium as a RPA tool for process automation, one of the objectives of this research.

Based on the stages described in Table 18, Figure 7 displays the structure of this CS.

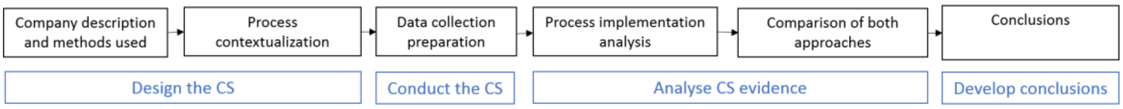

Company description and methods used
Process contextualization
Data collection preparation
Process implementation analysis
Comparison of both approaches
Conclusions
Design the CS
Conduct the CS
Analyse CS evidence
Develop conclusions

Figure 7 – CS Structure

For validating the CS, the author will follow the four tests proposed by Yin (2009), to ensure the quality of this research.

Table 19 - Validity tests (adapted from Yin (2009))

Test	Description
Construct Validity	*Multiple sources of evidence will be used (semi-structured interviews, observation and process documentation).*
Internal Validity	*This test is only suitable for explanatory studies. Since this CS is exploratory, this test will not be addressed.*
External Validity	*Analyzing the literature to define the domain, comparing the results of this CS with the related work. As there is no related work about using a Test Automation tool for RPA, this research will be a pioneer on this matter.*
Reliability	*The guidelines proposed by Yin (2009) will be followed, creating a path that shows how the investigation will be led, to ensure that the procedures can be repeated by future researchers and get similar results.*

Chapter 5 – Design the CS

This research was conducted in a company operating in the energy sector, which produces, distributes and sells electricity. It has more than 10000 employees in 4 continents, operating in more than 10 countries and serving more than 10 million clients. This research took place in the Information Systems department. This company created a RPA Center of Excellence, that helps creating a roadmap, gain knowledge, disseminate the best practices about RPA and define priorities and projects. Many processes have been automated over time. In 2017, more than 350000 euros were saved by automating processes with RPA. This company uses UiPath tool to automate processes, so it will be the RPA chosen tool in this research.

To answer the defined research questions, a process was automated, but first it was necessary to gather candidates for automation and choose the most suitable process for this research. Once the process was selected, it was necessary to analyse it, elicit the activities, cost, periodicity, systems involved, among others. Then implementation took place, automating the chosen process with Selenium and UiPath, followed by an analysis of the implementation with each tool and a comparison between them. To do so, this CS took place between October 2018 and March 2019. Table 20 presents this CS timeline.

Table 20 - CS timeline

	2018			**2019**		
	10	**11**	**12**	**1**	**2**	**3**
Process selection	■					
Chosen process analysis		■				
Implementation			■	■	■	
Implementation analysis						■

To conduct this CS, several methods to gather data were used, to ensure triangulation of information.

- Process documentation – essential to analyse the process;
- Semi-structured interviews – used to select the most suitable process, gather information about the selected process and collect the implementation findings;
- Observation – analyze the outcomes of the implementation, by looking at Selenium and UiPath interactions with elements on the browser. This can be useful for explaining differences between execution times in each activity using Selenium and UiPath, for instance.

After describing the methods used and defining the methods, a process contextualization is provided in the next sections, in terms of process selection and process description.

5.1. Process Selection

To select the most suited process for this research, a set of interviews were held. First, there was an interview with the people encharged for Architectures and Operations areas in order to identify the needs of each area in terms of automation. Then, once the processes were identified, individual interviews with each Process Owner were conducted to gather more information about each process.

After the interviews, several criteria were taken into consideration to select the most suited process, such as being suited for RPA automation ("RPA Suitable Process Criteria" section) and having as much as possible web automation, since Selenium only supports web automation applications. Table 21 presents the comparison between each process, using the criteria described above.

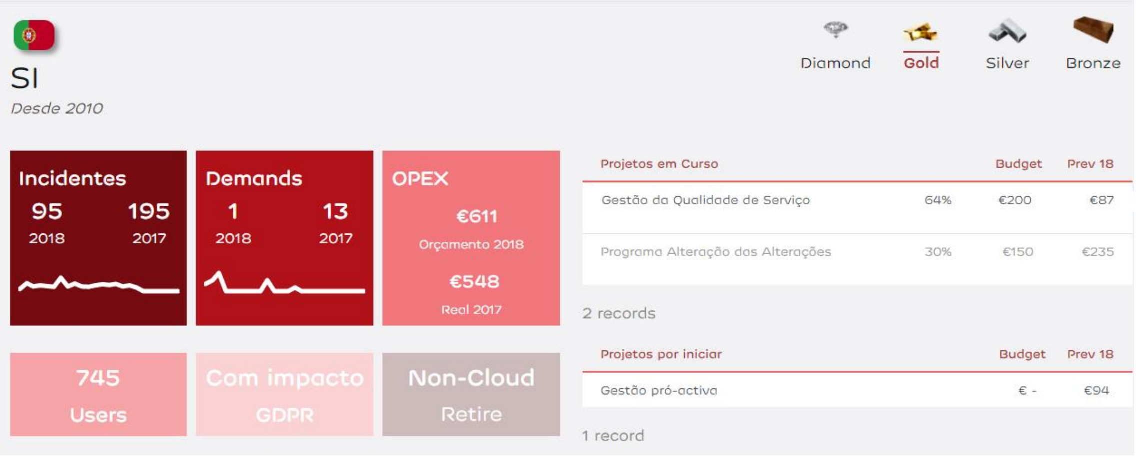
By comparing each process against the criteria defined, one can see that the "Upload of files to an APM tool" is the process that fulfils most of the criteria defined in terms of RPA process suitability, fulfilling 7 out of 13, whereas the "Software Agents Installation" process only fulfils 6 out of 13 and the "Infrastructure Consumption Validation" only fulfils 4 out of 13 criteria. Comparing each process in terms of Selenium suitability, one can check that the "Upload of files to an APM tool" process uses more websites than the other processes. The other two processes ("Software Agents Installation" and "Infrastructure Consumption Validation") have not much web automation and since the purpose of this research is to find out if a Test Automation tool can be used for RPA and Selenium is only capable of interacting with websites, they were not chosen to be automated in this research. The "Upload of files to an APM tool" is the chosen process, since it fulfils more criteria than the other two processes and interacts with more websites, being appropriate for this research.

Table 21 - Comparison between each process in terms of criteria

RPA Suitable Processes criteria	**Software Agent Instalation**	**Infrastructure Consumption Validation**	**Upload of files to an APM tool**
Voluminous transactions		*X*	
Frequent interaction with multiple systems	*X*	*X*	*X*
Use of systems with a stable environment			
Ease of decomposition into unambiguous rules	*X*		*X*
No need or limited worker intervention	*X*		
Limited need to handle exceptions	*X*		*X*
Awareness of current costs			*X*
Tasks prone to human errors		*X*	
High process maturity			*X*
High level of process standardization			
High quality of data			*X*
Low need of cognitive requirements	*X*		
High availability of digital data	*X*	*X*	*X*
Selenium suitability			
Number of websites used	*0*	*1*	*3*

5.2. Process AS-IS

As concluded in the previous section, “Upload of files to an APM tool” was the chosen process, which belongs to the Architectures area. This tool contains information about each application in terms of users, incidents, demands, projects, among others. The overview of each application serves to make decisions, whether it is to discontinue the application because it has a lower number of users and has many incidents or ask for more financing to improve the application, for instance. This information is gathered from many systems, combined into other files and then uploaded manually. As there are many sources of information and many files, it takes a long time to prepare the files to be uploaded. The information on the website must be updated regularly, so that decisions can be made with the most recent information. Figure 8 shows the APM tool dashboard for application “SI”. This screenshot and all websites screenshots further ahead are in Portuguese, as there was no English version available.



SI

Desde 2010

Incidentes
95 2018
195 2017

Demands
1 2018
13 2017

OPEX
€611 Orçamento 2018
€548 Real 2017

745 Users

Com impacto GDPR

Non-Cloud Retire

Projetos em Curso		Budget	Prev 18
Gestão da Qualidade de Serviço	64%	€200	€87
Programa Alteração das Alterações	30%	€150	€235

2 records

Projetos por iniciar	Budget	Prev 18
Gestão pró-activa	€ -	€94

1 record

Figure 8 - APM tool application dashboard

After describing the process, the subprocesses were identified. Figure 9 presents a Business Process Model and Notation (BPMN) diagram with the main subprocesses, which will be described in the following sub-sections in further detail.

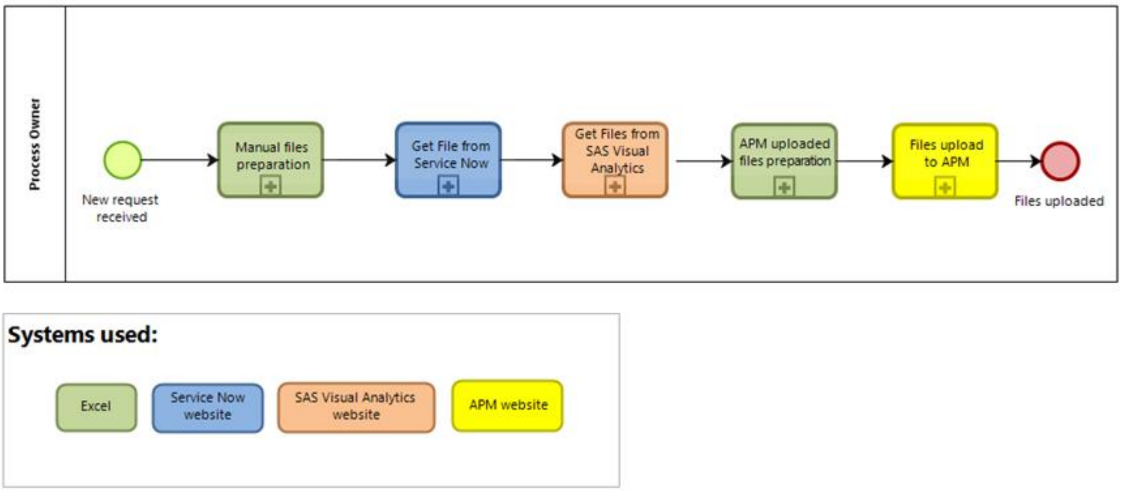


Figure 9 - Process BPMN diagram

In order to execute this process, several systems are used, such as websites and Excel. The websites used are Service Now, SAS Visual Analytics and the APM website (developed in OutSystems). The files (excel files) are retrieved from Service Now and SAS Visual Analytics, combined into other files (6 files, which are Applications, Incidents, Projects, Demands, Problems and Areas/Subareas) by getting the most appropriate attributes related to each application and then the 6 files are uploaded to APM website. All files are stored in a OneDrive folder.

The automation of this process may increase the speed and regularity of preparation of files to be upload to this tool, which can improve decision-making, as information provided in APM is more recent and accurate. By automating this process, the employee will have more time to focus on more important tasks.

After this contextualization, information was gathered in terms of time that this process takes, number of people that execute this process, periodicity of execution, cost per hour, where this process fits on the organization’s main processes and which processes this process interacts with. Table 22 presents some information about the process.

Table 22 - Information about the process

Time spent	Periodicity	Number of people	Cost per hour	Total cost
15 hours	*By request*	*1*	*45 €*	*630 €*

To get the files, combine them into other files, so that they can be uploaded, it takes 15 hours and costs 45€ per hour. One person executes this process when it is requested to update the information on the website.

This organization’s processes are organized according to Information Technology Infrastructure Library (ITIL), divided in 5 areas: Service Strategy, Service Design, Service Transition, Service Operation and Continuous Service Improvement. This process is situated in Service Strategy, in Application Porfolio Management and interacts with the following processes:

- Service Strategy – Demand Management and IT Financial Management;
- Service Design – Security Incident Management;
- Service Transition – Configuration Management and Project Management;
- Service Operation – Access Management, Incident Management, Problem Management and Service Request Management.

This process interacts with several processes (as described above), because each application contains information about its projects, demands, incidents, among others.

After identifying the main subprocesses, the time that each subprocess takes was estimated in terms of best-case, most likely and worst-case estimate and the estimation was calculated using the average of each value. Table 23 shows the estimation of each subprocess.

Table 23 - Estimation of each subprocess in hours/minutes

Subprocess	**Best-case (hour/min)**	**Most likely (hour/min)**	**Worst-case (hour/min)**	**Average (hour/min)**
Manual files preparation	*6h*	*7h*	*10h*	*7h36m*
Get File from Service Now	*0h5m*	*0h10m*	*0h20m*	*0h12m*
Get files from SAS Visual Analytics	*0h15m*	*0h25m*	*0h40m*	*0h27m*
APM uploaded files preparation	*6h*	*6h30m*	*8h*	*6h48m*
Files upload to APM	*0h3m*	*0h5m*	*0h10m*	*0h6m*

As can be seen, the preparation of manual files and the preparation of the files to be uploaded in APM are the subprocesses that take more time. The other subprocesses only take minutes to complete.

The following sub-sections contain a detailed description of each subprocesses.

5.2.1. Manual Files Preparation

There are some files that need to be prepared manually. These files have to be prepared in advance, so that some of their attributes can be used on building the files that are going to be uploaded to the APM tool.

Their names are:

- 6R – contains the migration to the cloud strategy for each application (Rehost, Replatform, Repurchase, Refactor, Retire and Retain);
- Budget of current year – contains for each application the budget for the current year in terms of operation, maintenance, production support and software licensing;
- Invoice of previous year – contains for each application the invoice in terms of operation, maintenance, licencing and production support;
- Seniority – the seniority year of each application;
- Areas/subareas – the areas and subareas of each application.

These files characterize the application in terms of its area, subarea, seniority, budget, invoice and satisfaction survey.

5.2.2. Get File from Service Now

There is an excel file that is available on the Service Now Configuration Management Database (CMDB) for download. This file is also related to each application and has many attributes, such as the status, application type, cloud level, accountable area, criticality and GDPR related data.

To download the file from Service Now, one must do the steps represented in Figure 10.

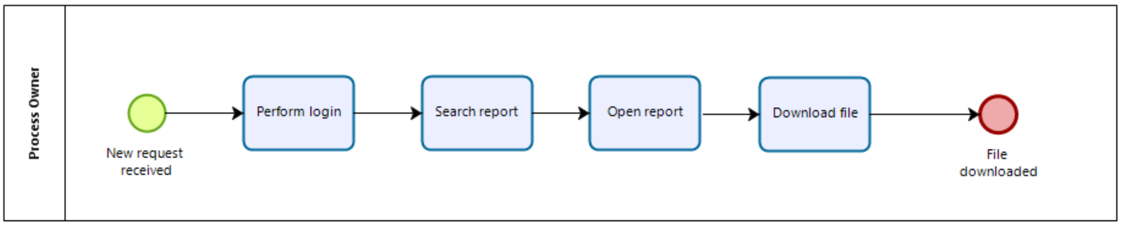


Figure 10 - Subprocess "Get file from Service Now" BPMN diagram

The first step is to login in Service Now. Then it is necessary to search the report and once it is found, open it in order to download it. To illustrate some of the described steps, Figure 11 displays screenshots for most steps.

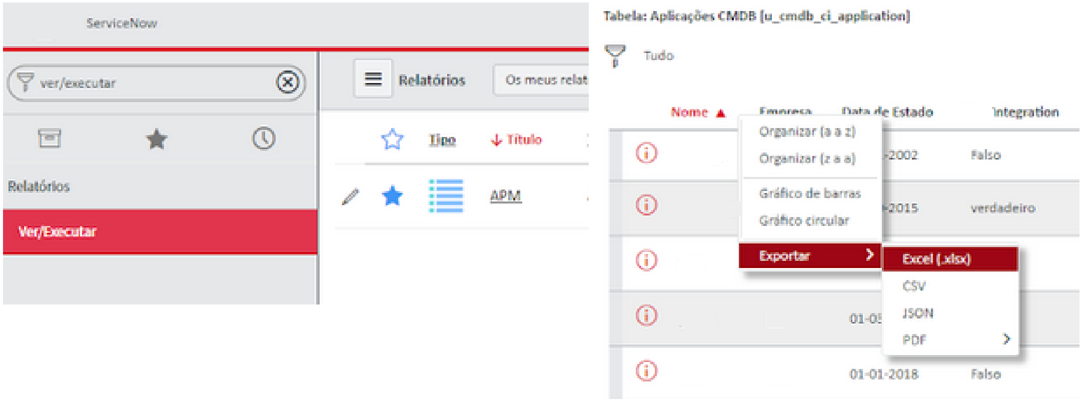


Figure 11 - Screenshots of the subprocess "Get file from Service Now"

5.2.3. Get Files from SAS Visual Analytics

There are several files (11 excel files), also related to each application for download in SAS Visual Analytics. They describe for each application and month the incidents reported, projects, demands and problems.

To download the files from SAS Visual Analytics, the steps are represented in Figure 12.

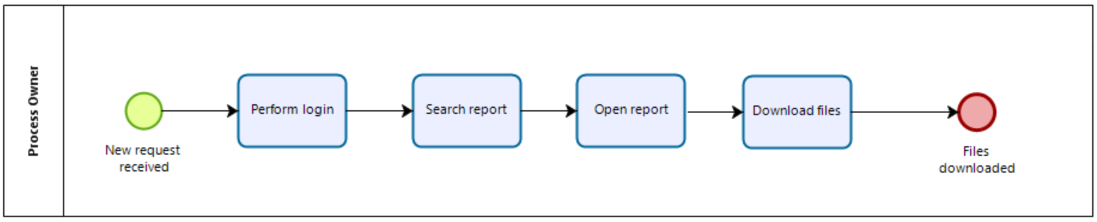
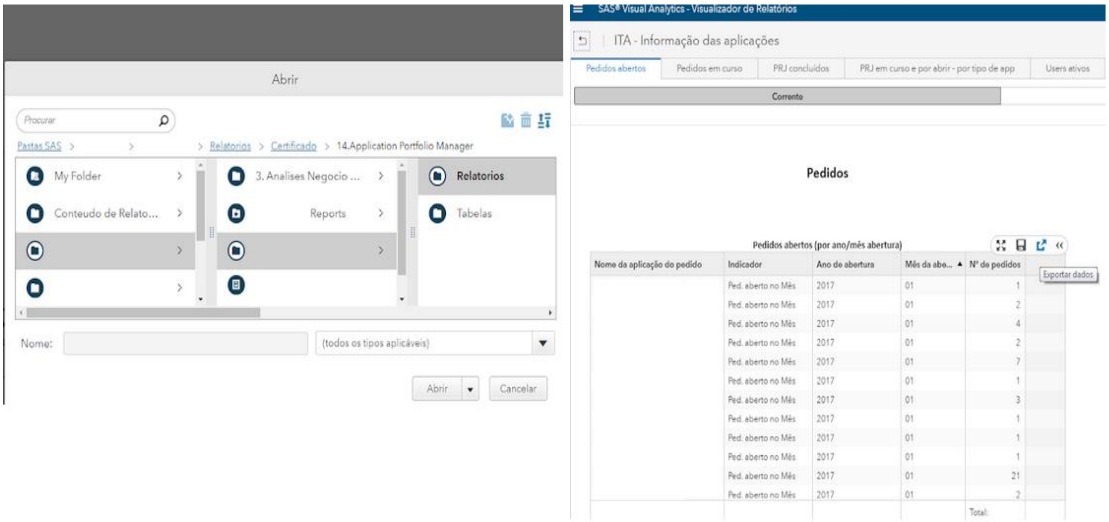


Figure 12 - Subprocess "Get file from SAS Visual Analytics" BPMN diagram

The first step is to login in SAS Visual Analytics. Then it is necessary to search the report and once it is found, open it. There are several tabs, each with a different file to download. To illustrate some of the described steps, Figure 13 displays screenshots for most steps.


Abrir
Procurar
Pastas SAS > > > Relatorios > Certificado > 14.Application Portfolio Manager
My Folder
Conteudo de Relato...
3. Analises Negocio ...
Reports
Relatorios
Tabelas
Nome:
(todos os tipos aplicáveis)
Abrir
Cancelar
SAS® Visual Analytics - Visualizador de Relatórios
ITA - Informação das aplicações
Pedidos abertos
Pedidos em curso
PRJ concluidos
PRJ em curso e por abrir - por tipo de app
Users ativos
Corrente
Pedidos
Exportar dados

Pedidos abertos (por ano/mês abertura)

Nome da aplicação do pedido	Indicador	Ano de abertura	Mês da abe... ▲	Nº de pedidos
	Ped. aberto no Mês	2017	01	1
	Ped. aberto no Mês	2017	01	2
	Ped. aberto no Mês	2017	01	4
	Ped. aberto no Mês	2017	01	2
	Ped. aberto no Mês	2017	01	7
	Ped. aberto no Mês	2017	01	1
	Ped. aberto no Mês	2017	01	3
	Ped. aberto no Mês	2017	01	1
	Ped. aberto no Mês	2017	01	1
	Ped. aberto no Mês	2017	01	1
	Ped. aberto no Mês	2017	01	21
	Ped. aberto no Mês	2017	01	2
				Total:

Figure 13 – Screenshots of the subprocess “Get file from SAS Visual Analytics”

5.2.4. APM Uploaded Files Preparation

After downloading the necessary files from Service Now and SAS Visual Analytics and storing them in a OneDrive folder, it is necessary to get the relevant attributes from each file and merge it into the appropriate file. Figure 14 displays the preparation of files to upload to APM.

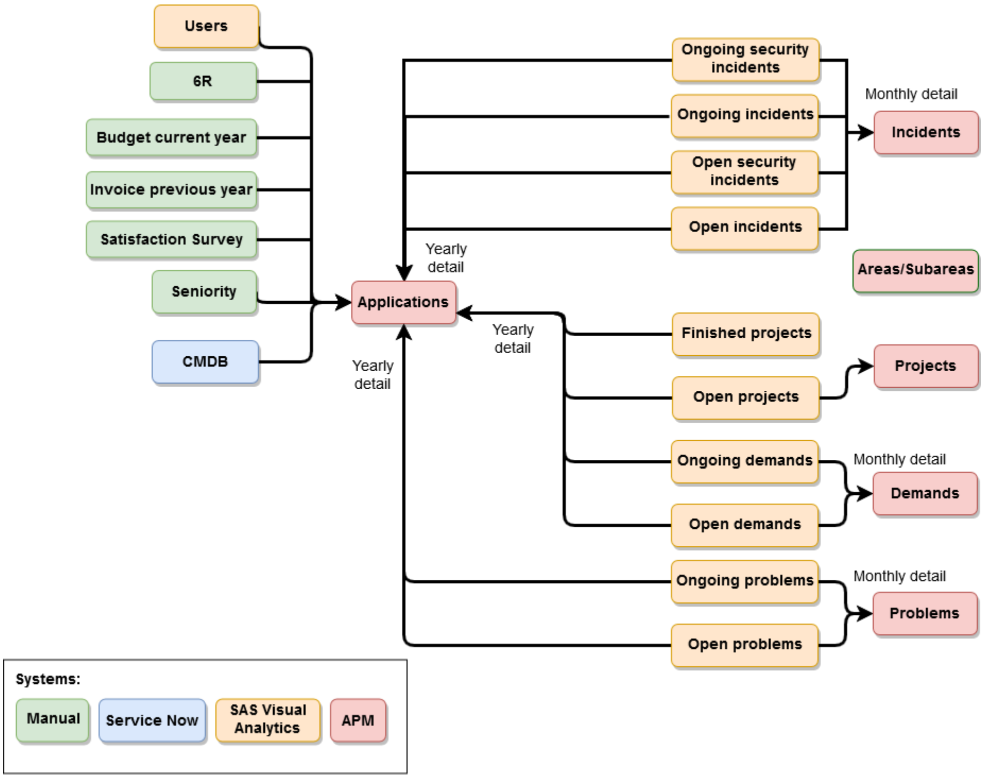


Figure 14 - Preparation of files to upload to APM

There are 6 files that are uploaded to APM, represented in Figure 14 in red. They are Applications, Incidents, Projects, Demands, Problems and Areas/Subareas. The green boxes are manual files that are already prepared (as explained in section "Manual files preparation". The blue and orange boxes are the files downloaded from Service Now and SAS Visual Analytics. In each file (green, yellow and blue boxes), the appropriate attributes regarding each application are stored in each file (red boxes) and then the files in the red boxes are uploaded to the APM website. The files to be uploaded to APM website are prepared from the following files:

- Application – get the appropriate attributes from Users, 6R, Budget current year, Invoice previous year, Satisfaction Survey, Seniority, Areas/Subareas and CMDB files. Plus, it stores the number of incidents (ongoing and open), projects (finished and open), demands (ongoing and open) and problems (ongoing and open) of the previous year and the current year for each application, each in a separate column;
- Incidents – gets all attributes of incident files (ongoing security incidents, ongoing incidents, open security incidents and open incidents), providing for each application the number of incidents per month and year;
- Projects – gets all attributes of open projects file for each application;
- Demands - gets all attributes of demand files (ongoing demands and open demands), providing for each application the number of demands per month and year;
- Problems - gets all attributes of demand files (ongoing problems and open problems), providing for each application the number of problems per month and year;
- Areas/Subareas – this file is prepared manually, so it is ready for upload to APM without requiring further preparation.

5.2.5. Files Upload to APM

After preparing the files to be uploaded, the upload of files to APM takes place.

To upload the files (Applications, Incidents, Projects, Demands, Problems and Areas/Subareas), the following steps are required, as represented in Figure 15.

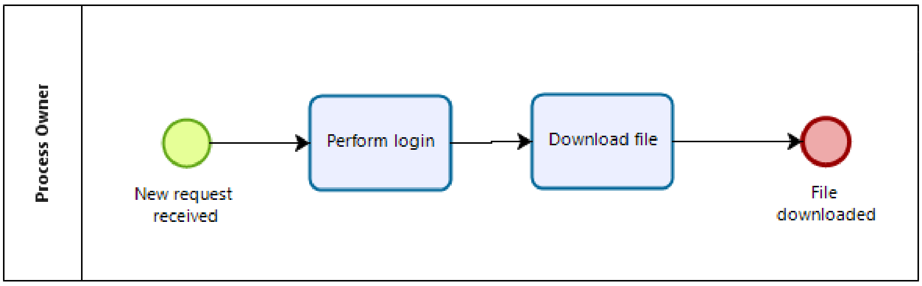
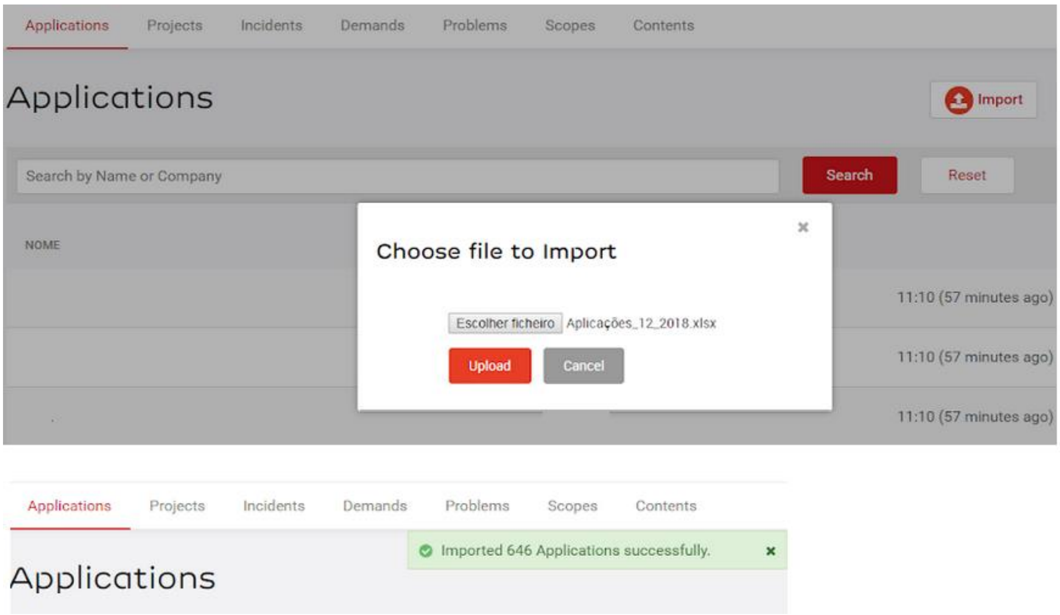


Figure 15 - Subprocess "Files upload to APM" BPMN diagram

This subprocess starts by logging into APM. Then several tabs are shown, each tab for each file. By pressing the upload button in each page, it uploads the appropriate file, displaying a message after the upload is complete. To illustrate the some of the described steps, Figure 16 displays screenshots for most steps.



Choose file to Import

Aplicações_12_2018.xlsx

11:10 (57 minutes ago)

11:10 (57 minutes ago)

11:10 (57 minutes ago)

Imported 646 Applications successfully.

Figure 16 – Screenshots of the subprocess "Files upload to APM"

The next section presents the automation strategy, identifying the subprocesses to automate and required justification.

5.3. Automated Process

Before automating the process, it is necessary to identify which subprocesses can be automated, as some subprocesses may require human intervention. Figure 17 presents the BPMN diagram with the main subprocesses. The process will be executed by request.

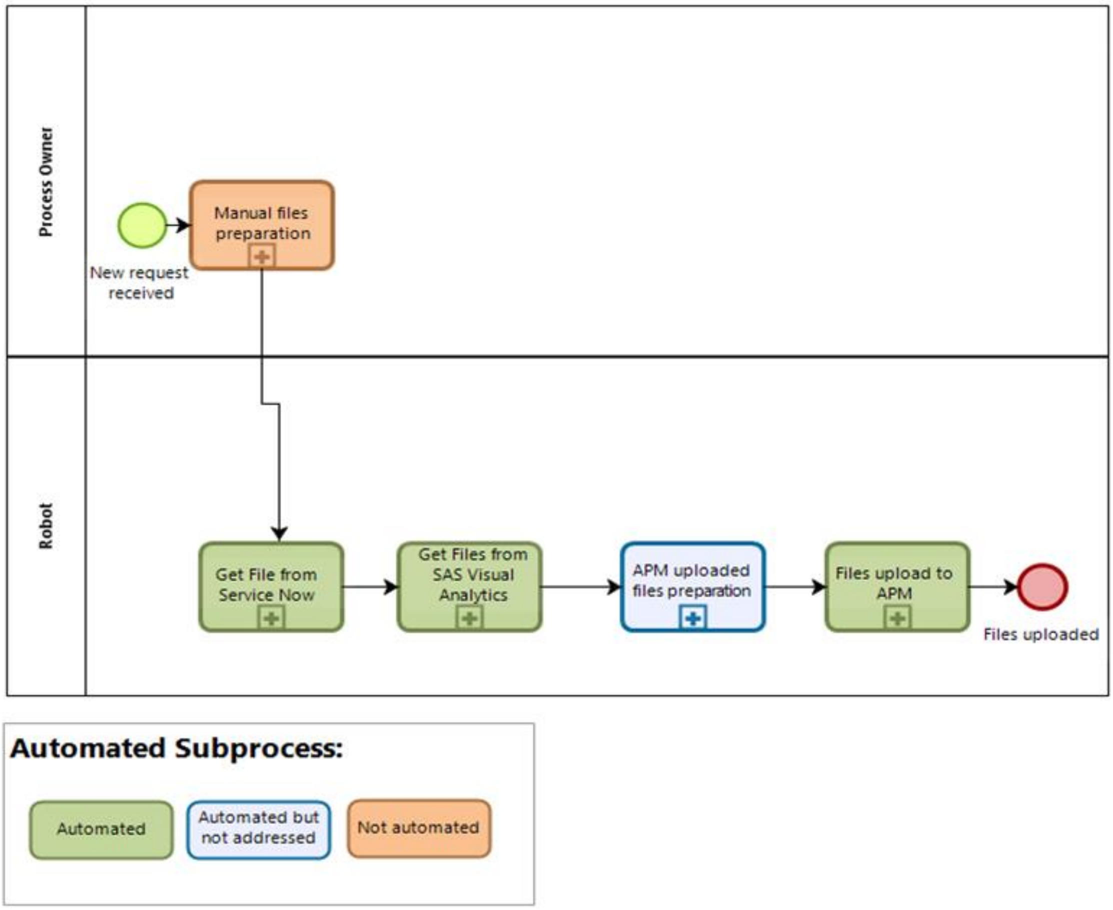


Figure 17 – Automated process BPMN diagram

The subprocess “Manual files preparation” will not be automated, since it requires worker intervention to prepare the files. The remaining subprocesses will be automated using Selenium and UiPath, except for the subprocess “APM uploaded files preparation”. As Selenium only focuses on web automation activities, is unable to automate this task (“APM uploaded files preparation”), because there is data manipulation involved and no interaction with websites. Despite automated, this subprocess will not be used to compare Selenium against UiPath.

Each subprocess will be automated in Selenium Webdriver first. Selenium Webdriver is one of Selenium tools, supporting many languages to write scripts. It was chosen Python language. Selenium Webdriver with Python was chosen because one of the subprocesses needed data manipulation (“APM uploaded files preparation”) and as Selenium IDE only supports web automation, it was necessary the help of a language to overcome that obstacle and as Selenium Webdriver is used within a programming language, it was the chosen tool. After each subprocess is automated in Selenium Webdriver, they will be automated using UiPath. Both approaches will be compared (automating in Selenium Webdriver and UiPath) and the conclusions will help answering the research questions defined.

Chapter 6 – Conduct the CS

After the company and process contextualization and definition of methods, the CS conduction took place. For the conduction of this CS, Table 24 shows the interviewees characterization.

Table 24 - Interviewees characterization

Interviewee	Role	Area
A	*Architectures responsible*	*Architectures*
B	*Operations responsible*	*Operations*
C	*Process owner of "Software agent installation"*	*Operations*
D	*Process owner of "Infrastructure consumption validation"*	*Operations*
E	*Process owner of "Upload of files to an Application Portfolio Management (APM) tool"*	*Architectures*
F	*RPA Developer of the chosen process*	*Development*

To select the most suited process, there was an interview with the Architectures and Operations responsibles (A and B interviewees) to identify the needs of each area in terms of automation. Once the processes were identified, individual interviews with each process owner (C, D and E interviewees) were conducted to collect for information about the process. Once the chosen process is automated with Selenium and UiPath, an interview with the developer (F interviewee) was conducted in order to identify the difficulties and workarounds of automating a process using Selenium and UiPath.

To resume this chapter, Table 25 displays for each phase identified in Table 20, which methods to gather data were used and the participants involved.

Table 25 - Methods used and participants involved in each CS phase

CS phase	Methods to collect data	Participants involved
Process selection	*Semi-structured interviews*	*A, B, C, D, E*
Chosen process analysis	*Process documentation*	*E*
Implementation	*NA*	*F*
Implementation analysis	*Observation* *Semi-structured interview*	*F*

Chapter 7 – Analyse CS Evidences

To get and then analyse CS evidence, an interview with the developer who implemented this process using Selenium and then UiPath was conducted in order to describe how each activity was implemented and identify the problems of implementation and workarounds needed. Also, there was an observation of the process execution running, to explain some of the findings. The results are used to compare both approaches (Selenium and UiPath) and then answer the research questions defined in the research. The developer interviewed implemented the process first using Selenium and then UiPath. He has a Computer Engineering degree and has no experience in using Selenium Webdriver with Python or UiPath.

In the following sections, for each technology and activity, there is a brief description of each activity, the difficulties identified, and the solutions found to solve them. Before moving on to the next sections, it is necessary to identify some terminology. To get the elements present on the screen of a website and then interact with them, both Selenium and UiPath have to identify each element (a button, a text box, for instance) by looking at the HyperText Markup Language (HTML) of the website and then uniquely identify the element. Each element on the HTML has its own tag (an element name within angle brackets, like <a> or <h1>) and inside the tag there may be attributes (properties that modify and caractherize an element, like an id or class) (Robbins, 2012). To identify an element uniquely one must use a tag or attribute or a combination that can uniquely identify that element, so that Selenium/UiPath can interact with the correct element. Figure 18 represents an element with the respective tag and some attributes.

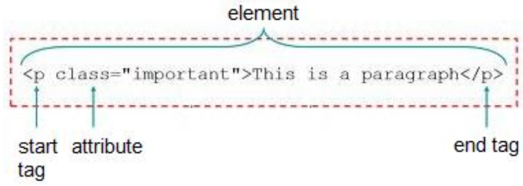


Figure 18 - HTML element structure

This terminology (element, tag and attribute) is used throughout the next sections.

7.1. Selenium

The chosen Selenium tool in this research is Selenium Webdriver. Selenium Webdriver is used as a library of a programming language. In this research, the chosen tool was Python. To communicate with the browser, Selenium Webdriver uses the appropriate driver (each browser has a different driver) to send and receive commands (Figure 19).

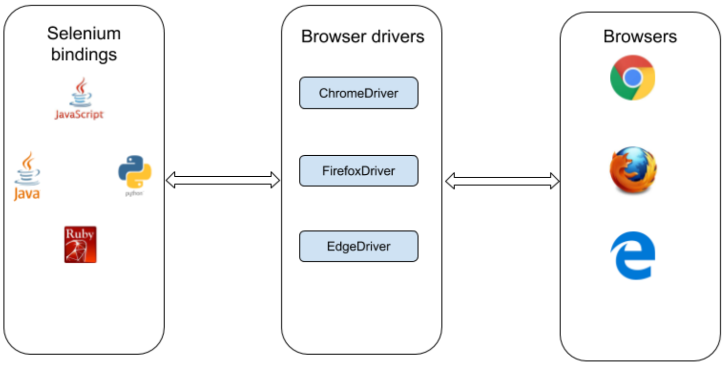


Figure 19 - Selenium Webdriver architecture

To identify and interact with the elements, Selenium uses locators to find the elements, such a name, id, class, XPath, among others (Unmesh, 2012). To get the element, one must investigate the HTML code manually and pick the appropriate attribute or a combination of attributes and then use the appropriate locator method, in which Selenium Webdriver will use to interact with the element. Another way is to use Selenium IDE, one of the Selenium tools, to record the interactions with the elements and capture the attributes that identify uniquely the element and use them on Selenium Webdriver, but Selenium IDE sometimes identifies the element using the wrong attributes, so in that case it is also necessary to look at the HTML.

To organize the code, the developer used a design pattern for Selenium called Page Object. This code pattern makes the code more maintainable, hiding the implementation from the tests (Vila et al., 2017). Each webpage has its own class, where the methods and locators needed to interact with that page are defined and the test calls the necessary methods of the webpages (Unmesh, 2012). For instance, if one wants to automate a login page, one must create a class that has locators of each textbox (username and password) and submit button and the methods to login, given a username and password and the main class (test class) uses the methods to login.

The next subsections present the problems and solutions identified for each subprocess implemented in Selenium.

7.1.1. Get File from Service Now

In this subprocess, one had to login into Service Now, search the report, click on it and then download it. Throughout the implementation, there were some difficulties encountered. Table 26 displays such problems and workarounds.

There was a recurring problem in all subprocesses that were implemented in Selenium. The HTML of the used webpages had iframe tags (a HTML document embedded in other HTML document) and some elements were inside the iframe tags (P1). When the developer wanted, for instance, click on a button, he had to find the appropriate attribute that can uniquely identify the element. Once he got that attribute, when he runned the program, an exception has thrown (element not found). After researching, he reached the conclusion that the element was inside an iframe and it was needed to first get the attribute that identifies the iframe, use a method to change to the context of that iframe and then use the identified attribute for the element (W1).

Table 26 - Problems and workarounds of subprocess "Get file from Service Now"

Id	Problem	Id	Workaround
P1	*Could not find the elements because they were inside the iframe tag*	*W1*	*Get the attribute that identifies the iframe and use the Selenium method to switch to that iframe, so that elements can be found*
P2	*Capture elements with attributes that are always changing*	*W2*	*Find a combination of static attributes that can uniquely identify the element*
P3	*Different parts of the webpage had different loading times*	*W3*	*Delay so that the page is completely loaded*
P4	*Cannot click on elements that are disabled*	*W3*	*Delay so that the element can become clickable*

In Service Now webpage, there were some difficulties to get a combination of attributes that could uniquely identify the element (P2), because some attributes (like the id attribute) kept on changing dynamically and other ways to identify attributes had to be used, like using xpath (W2).

Another problem encountered was the fact the different parts of the webpage had different loading times (P3). An example of that is in Figure 20.

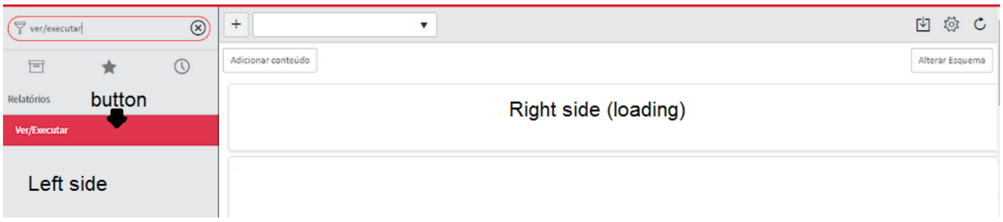
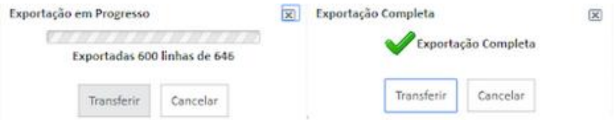


Figure 20 – Different parts of the webpage with different loading times problem

When someone wants to go to the reports area, has to search and then click on the search result, but the right side of the webpage (initial page) is still loading. When the search result is clicked, sometimes it was ignored because the right side was still loading. By inserting a delay before clicking on the search result button allowed the right page to fully load (W3).

After going to the report, one must download the report. After that a popup appears with a disabled download button (P4) but after finishing loading, the button is clickable (Figure 21). To solve that, it was necessary to insert a delay until the button was clickable (W3).


Exportação em Progresso

Exportadas 600 linhas de 646

Exportação Completa

Exportação Completa

Figure 21 - Can not click on elements that are disabled problem

7.1.2. Get Files from SAS Visual Analytics

To execute this subprocess, it is necessary to login on SAS Visual Analytics, search the report page and for each tab on that page download the respective file. While implementing this subprocess, there were some problems that occurred, represented in Table 27.

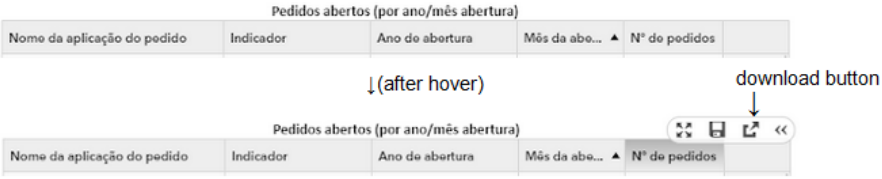
Table 27 - Problems and workarounds of subprocess "Get files from SAS Visual Analytics"

Id	Problem	Id	Workaround
P5	*The website was down, or the reports were not available*	*W4*	*Delay until the problem is solved and then run the robot again*
P1	*Could not find the elements because they were inside the iframe tag*	*W1*	*Get the attribute that identifies the iframe and use the Selenium method to switch to that iframe, so that elements can be found*
P6	*Difficult to hover toolbar, open it and click on an element inside it*	*W3*	*Delay opening the toolbar and once the toolbar is loaded click on an element*
P7	*Click on elements (tabs) invisible on the screen*	*W5*	*Click on a button that makes the other tabs appear on the screen*

This website sometimes was under maintenance or the reports were not available, so when the robot tried to download them, they were not available and could not be downloaded (P5), so the execution had to be aborted and when the reports were available again, the robot could run properly (W4).

As the previous subprocess, there were also problems with the iframes (P1). The solution was to get the attribute that identifies the iframe, switch to that iframe and interact with the elements inside of it (W1).

When a report was downloaded, one must hover around an invisible element and then a toolbar opens with a download button (P6) (Figure 22). As the toolbar took some seconds to load, it was necessary to insert a delay to wait until the toolbar is loaded, so that the download button inside of it could be clicked, without throwing an exception (W3).



Pedidos abertos (por ano/mês abertura)

Nome da aplicação do pedido	Indicador	Ano de abertura	Mês da abe... ▲	Nº de pedidos	

Pedidos abertos (por ano/mês abertura)

Nome da aplicação do pedido	Indicador	Ano de abertura	Mês da abe... ▲	Nº de pedidos	

Figure 22 - Difficult to hover mouse and click on elements problem

The last problem was clicking on tabs that were not visible on the screen (P7). As the page had 11 tabs, depending on the size of the screen, some tabs were not shown on the screen, despite being loaded. The tabs that were not shown on screen could not be clicked, so in order to click on them, one must press the (>) button to show the tabs (W5) (Figure 23).

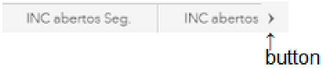


Figure 23 - Click on tabs that are not visible on screen problem

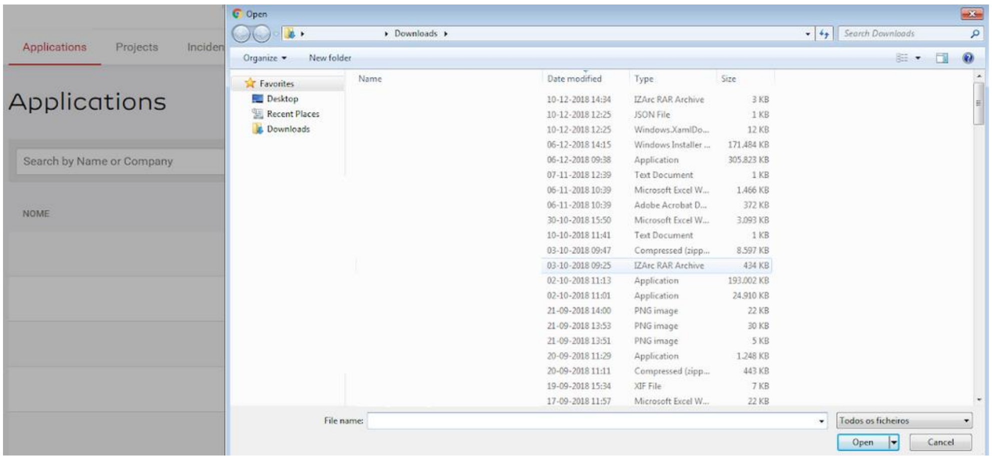
7.1.3. Files Upload to APM

This subprocess is concerned with uploading 6 files to the APM website, going to each tab on the website, uploading the appropriate file and once the file is uploaded, moving on to the next file and so on. During the implementation, there were some problems encountered, represented in Table 28.

Table 28 - Problems and workarounds of subprocess "Files upload to APM"

Id	Problem	Id	Workaround
P8	*Selenium cannot access elements outside the browser scope, such as the file selection window*	*W6*	*Instead of clicking on the upload widget to open the file selection window, send the location of the file to the file selection input field without opening the file upload window*
P9	*When elements are temporarily in front of other elements, these other elements cannot be clicked*	*W3*	*Delay until the front element disappears and then click on the other element*

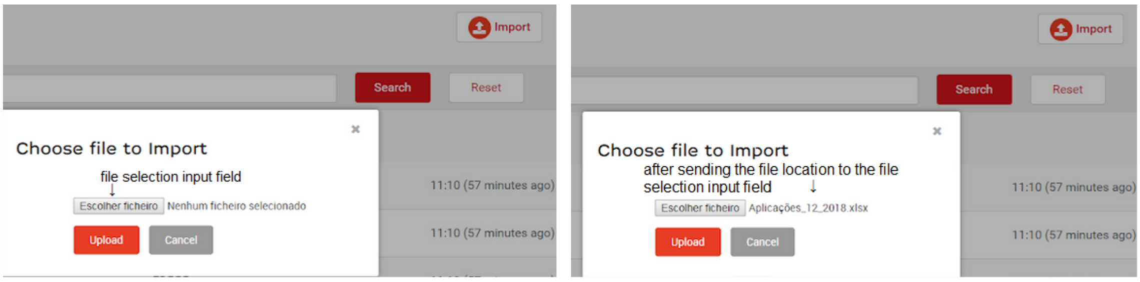
As Selenium interact only with webpages, it cannot access elements outside it (P8), such as the file selection window, represented in Figure 24.



Name	Date modified	Type	Size
	10-12-2018 14:34	IZArc RAR Archive	3 KB
	10-12-2018 12:25	JSON File	1 KB
	10-12-2018 12:25	Windows.XamlDo...	12 KB
	06-12-2018 14:15	Windows Installer ...	171.484 KB
	06-12-2018 09:38	Application	305.823 KB
	07-11-2018 12:39	Text Document	1 KB
	06-11-2018 10:39	Microsoft Excel W...	1.466 KB
	06-11-2018 10:39	Adobe Acrobat D...	372 KB
	30-10-2018 15:50	Microsoft Excel W...	3.093 KB
	10-10-2018 11:41	Text Document	1 KB
	03-10-2018 09:47	Compressed (zipp...	8.597 KB
	03-10-2018 09:25	IZArc RAR Archive	434 KB
	02-10-2018 11:13	Application	193.002 KB
	02-10-2018 11:01	Application	24.910 KB
	21-09-2018 14:00	PNG image	22 KB
	21-09-2018 13:53	PNG image	30 KB
	21-09-2018 13:51	PNG image	5 KB
	20-09-2018 11:29	Application	1.248 KB
	20-09-2018 11:11	Compressed (zipp...	443 KB
	19-09-2018 15:34	XIF File	7 KB
	17-09-2018 11:57	Microsoft Excel W...	22 KB

Figure 24 - File selection window

When one wants to select a file to be uploaded, he clicks on a file selection input field that opens a file upload window (outside of Selenium scope) to select the file location. As Selenium is unable to interact the file upload window, it has allowed to send text (in this case the location of the file) to file selection input field, meaning that it is not necessary to click on it and open the file upload window (W6) (Figure 25).

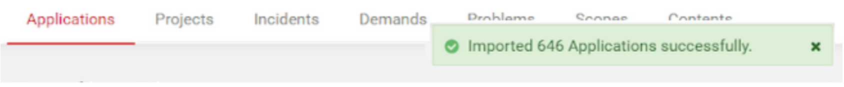


file selection input field

after sending the file location to the file selection input field

Figure 25 – Selenium solution for not opening the file selection window

The last problem occurred when a message appeared after the file was successfully uploaded. The message was in front of the tabs for some seconds and then disappeared (Figure 26). If one clicked on the next tab to upload the next file while the message was in front of the tab, an exception was thrown (P9), so clicking on the next tab once the message disappeared solved the problem (W3).

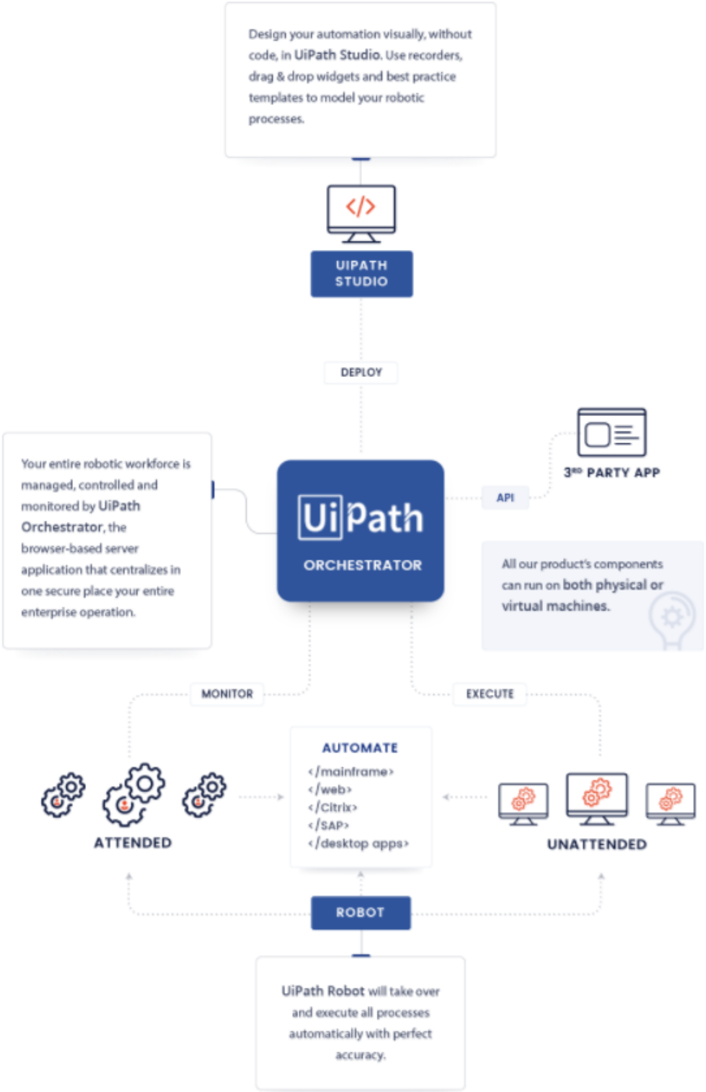


Imported 646 Applications successfully.

Figure 26 - Message in front of tabs

7.2. UiPath

UiPath was founded in 2005. It offers three products, which are UiPath Studio, UiPath Orchestrator and UiPath Robot (Figure 27).



Design your automation visually, without code, in UiPath Studio. Use recorders, drag & drop widgets and best practice templates to model your robotic processes.

Your entire robotic workforce is managed, controlled and monitored by UiPath Orchestrator, the browser-based server application that centralizes in one secure place your entire enterprise operation.

All our product's components can run on both physical or virtual machines.

UiPath Robot will take over and execute all processes automatically with perfect accuracy.

Figure 27 - UiPath Architecture (retrieved from UiPath website)

UiPath Studio provides the means to develop a robot without coding, just using drag and drop. Users can also develop in VB.NET or C#.NET to create new functionalities that can be re-used. UiPath Orchestrator is a server where robots can be orchestrated. UiPath Robot can be attended or unattended. On attended automation, robots can cooperate with users when their assistance is required, whereas on unattended automation, robots perform tasks on their own without the user's assistance (Tran & Ho Tran Minh, 2018).

To identify the elements and interact with elements on webpages, UiPath uses its recorder functionality, that can record mouse movements and pressed keys and then generates the sequence of steps recorded. Also, when recording, it captures the elements being interacted with by selecting the most appropriate attributes automatically. UiPath is also able to interact with desktop apps.

To organize the code, the developer arranged the activities by splitting each activity into a separate flowchart, represented in Figure 28. By double clicking each flowchart, one can insert a sequence of steps performed for the appropriate subprocess.

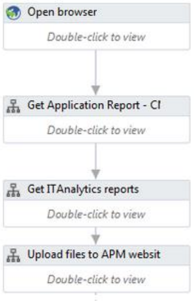


Figure 28 - Organization of code in UiPath

The next subsections present the problems and solutions identified for each subprocess implemented in UiPath.

7.2.1. Get File from Service Now

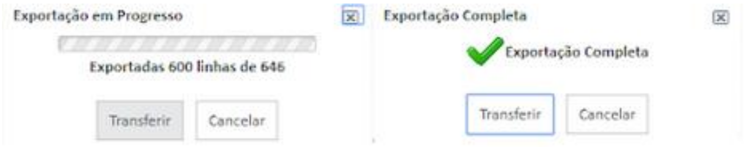
In this subprocess, one needed to go into Service Now, search the report, click on it and then download it. During the implementation of this activity, there were some problems reported. Table 29 displays the problems and workarounds of this subprocess.

Like Selenium, UiPath had problems with capturing elements with dynamic attributes (P2). Even though UiPath is able to record actions and capture the elements automatically, many times it did not pick the correct attributes to identify the element and when the program was executed, an exception (not finding the element) was thrown. To solve that problem, the developer had to try all combinations of attributes manually to find the one that did not throw an exception (W2).

Table 29 - Problems and workarounds of subprocess "Get file from Service Now"

Id	Problem	Id	Workaround
P2	*Capture elements with attributes that are always changing*	*W2*	*Find a combination of static attributes that can uniquely identify the element*
P10	*No exception thrown when clicks on elements that are disabled*	*W3*	*Delay so that the element can become clickable*

UiPath does not thrown an exception when it clicks on elements that are disabled (P10), unlike Selenium. By not throwing an exception, it is more difficult to understand why the execution is not running as intended. For instance, in this subprocess, when the download popup appears, the download button is disabled and once it has finished exporting, the download button is clickable (Figure 29). As the robot clicks on the download button, it is still disabled and the download does not begin, but as it does not give any feedback (an exception) to the developer, it is harder to detect the problem and find a solution. One must notice the behavior of the button and add a delay until the button is clickable in order to download the report (W3).


Exportação em Progresso

Exportadas 600 linhas de 646

Exportação Completa

Exportação Completa

Figure 29 - Can not click on elements that are disabled problem

7.2.2. Get Files from SAS Visual Analytics

For the robot to execute this subprocess, is is necessary to login on SAS Visual Analytics, search and open the report page and for each tab on that page download the respective file. Throughout the implementation, there were some problems that occurred, represented in Table 30.

This website was under maintenance or the reports were not available, so when the robot tried to download them, they were not available and could not be downloaded (P5), so the execution had to be aborted and when the reports were available again, the robot could run properly (W4).

Table 30 - Problems and workarounds of subprocess "Get files from SAS Visual Analytics"

Id	Problem	Id	Workaround
P5	*The website was down, or the reports were not available*	*W4*	*Wait until the problem is solved and then run the robot again*
P2	*Capture elements with attributes that are always changing*	*W2*	*Find a combination of static attributes that can uniquely identify the element*
P6	*Difficult to hover toolbar, open it and click on an element inside it*	*W7*	*Get the download button hidden on screen without hovering and click on it*

As the previous subprocess, there were also problems with capturing elements with attributes that are always changing (P2) and despite UiPath is able to record the actions and get the attributes that identify the element automatically, it sometimes does not find the best attributes that identify the element uniquely, so it is necessary to find a combination of attributes that do not change and can uniquely identify the element manually (W2).

The last problem was hovering a hidden element, so a toolbar to download each file can be opened (P6). As UiPath is able to click on elements that are invisible on the screen, the attributes that identify the download button were gathered and then the robot was able to click on that hidden element without opening the toolbar (W7).

7.2.3. Files Upload to APM

This subprocess is concerned with upload files to the APM website, by going to each tab, uploading the appropriate file and once the file is uploaded, moving one to the next file and so on. While implementing this subprocess, there were some problems encountered, represented in Table 31.

Table 31 - Problems and workarounds of subprocess "Files upload to APM"

Id	Problem	Id	Workaround
P11	*File selection input field was clicked but no file selection window appeared*	*W3*	*Delays for all elements on the upload popup to load and then click on the file selection input field on that popup*
P9	*When elements are temporarily in front of other elements, these other elements cannot be clicked*	*W3*	*Delay until the front element disappears and then click on the other element*

One of the problems identified was that when the file selection input field was clicked, no window to select the file location appeared (P11) (Figure 30). As UiPath did not throw an exception, the developer tried to insert delays between opening the popup and clicking on the file selection input field to see if that would work (W3). It worked and what probably happened was that the popup was still loading when the file selection input field was clicked, so no window to select the file location appeared.

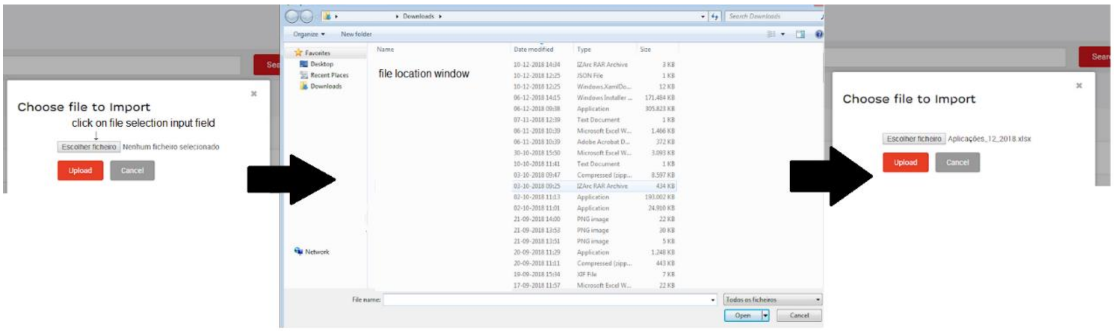
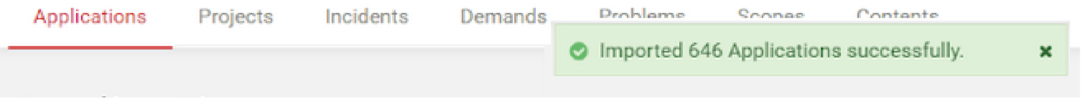


Figure 30 - Sequence of steps for uploading a file

The last problem happened when a feedback message appeared after the file was successfully uploaded. The message was in front of the tabs for some seconds and then disappeared (Figure 31). If the robot clicked on the next tab to upload the next file while the message was in front of the tab, the next page would not load, which meant that the tab was not clicked (P9). The solution was to click on the next tab once the message disappeared (W3).



Imported 646 Applications successfully.

Figure 31 - Message in front of tabs

7.3. Problems and Workarounds of Selenium and UiPath Overview

After reporting the problems identified, one can see that in Selenium there were more problems identified than in UiPath (Table 32), which is not surprising, as Selenium is used for a different purpose than automating processes. The subprocesses represented with numbers are: 1- Get file from Service Now, 2- Get files from SAS Visual Analytics and 3- Files upload to APM.

By looking at the correspondence between the problems reported and the subprocesses, one can see that in some cases the same subprocess has different problems in Selenium and UiPath. For instance, in the first subprocess, one only problem is common to both technologies (P2) but the remaining problems are different (Selenium with P1, P3 and P4, UiPath with P10). This means that as each technology is different and used for a different purpose (Selenium for test automation and UiPath for RPA), it is normal that different problems appear.

Table 32 - Overview of problems and workarounds

Problem	**Selenium**		**UiPath**	
	Subprocess	**Workaround**	**Subprocess**	**Workaround**
P1	*1,2*	*W1*	-	-
P2	*1*	*W2*	*1,2*	*W2*
P3	*1*	*W3*	-	-
P4	*1*	*W3*	-	-
P5	*2*	*W4*	*2*	*W4*
P6	*2*	*W3*	*2*	*W7*
P7	*2*	*W5*	-	-
P8	*3*	*W6*	-	-
P9	*3*	*W3*	*3*	*W3*
P10	-	-	*1*	*W3*
P11	-	-	*3*	*W3*

Despite the specificity of each technology, in some subprocesses there were common problems in Selenium and UiPath, such as P5 and P6 in the second subprocess, which demonstrates that both technologies share some problems. More examples of that are being difficult to identify the elements (P2) and difficulties in clicking on disabled elements (P3/P10). Both technologies reported problems with doing the same activities, such as uploading a file to APM website, click on the download button on Service Now to download the report and also clicking on the download button on SAS Visual Analytics. Adding delays (W3) in most activities solved the problems in both technologies.

Some problems were not reported in UiPath, but were in Selenium, such as P1. This happens because one of the problems of Selenium is that is only able to interact with elements on an iframe by getting the attribute that identifies the iframe and change to the context of the iframe in order to interact with elements. The opposite also happened (problems found in UiPath and not in Selenium). One example of that is P11. As UiPath is unable to pass the location of the file to be uploaded to the file selection input field without opening the file selection window like Selenium can, it struggled to open the file selection window, so delays had to be added.

Regarding the workarounds used, one can see that for both technologies they were the same, except for the problem of being difficult to hover toolbar, open it and click on a download button inside it (P6). In that case, as Selenium is unable to interact with elements that are invisible on screen, it could not click on the download button to download the SAS Visual Analytics files without opening the toolbar which contains the downlod button (W3). UiPath, on the other hand, could do that and was able to click on the download button without opening the toolbar (W7).

7.4. Comparison of Both Approaches

This section aims to compare Selenium with UiPath in general terms, such as their goal, interaction with applications, type of software, cost, operating systems and software requirements. Afterwards, based on the findings described in the previous section, both technologies are compared in terms of software skills, implementation time, implementation difficulty, execution time, maintainability and finally technical aspects (Table 35).

Starting with the general terms, as described in the Literature Review, Selenium seeks to do test automation, whereas UiPath is a RPA tool, aimed to automate processes. Selenium is a free and open source tool that is available for all operating systems and can only interact with webpages. To use Selenium Webdriver, it is necessary to install a programming language (in this case Python), get the Selenium Webdriver library for that language and also download the driver for the chosen browser, so that Selenium Webdriver can communicate with the browser. On the other hand, UiPath is a costly proprietary software that runs on Windows and can be used to automate web and desktop applications. To use UiPath, it is only necessary to download and install its software.

To use Selenium, it is necessary to have programming skills, as it is incorpored within a programming language (Python, in this case) and because of that, it takes more time to implement. UiPath is a drag and drop software, but still knowledge of logic is required. As there is no code and many functionalities are already implemented and can be used, it takes less time to implement. Table 33 displays the time that took to implement the process in Selenium and UiPath.

Table 33 - Implementation time in Selenium and UiPath

Selenium implementation time	UiPath implementation time
3 weeks	*1.5 weeks*

In terms of difficulty of implementation, as seen in the previous section, there were more problems detected during the implementation with Selenium than with UiPath, specially because of the fact that to get the elements, one must look into the HTML and find the combination of attributes that uniquely identify the element manually, whereas in UiPath it is done automatically (it captures the element automatically and then if the developer notices that the wrong attribute was chosen, he can pick other attribute easily). Moreover, as the activities in UiPath are represented in a visual manner, it is easier to detect bugs and correct them and therefore the maintainability is easier. On the other hand, Selenium requires coding, which is not visual, and it is harder to detect and correct bugs, being more difficult to maintain the code.

To compare Selenium with UiPath in terms of execution time, this process was executed 30 times in Selenium and UiPath and for each subprocess the execution time was registered. The results are in Table 34. The subprocesses represented with numbers are: 1- Get file from Service Now, 2- Get files from SAS Visual Analytics and 3 - Files upload to APM.

In terms of total execution times, in average Selenium took less time to execute (5 minutes and 3 seconds), whereas UiPath took more time (5 minutes and 15 seconds). The difference between the two technologies is not much. The small difference between the execution times can be analyzed in more detail by comparing the performance of the subprocesses in Selenium and UiPath. In the first subprocess (Get file from Service Now), Selenium took more time than UiPath (37 seconds for Selenium and 23 seconds for UiPath). As seen in the previous section, there were more problems in this subprocess using Selenium than with UiPath. The most importance difference between times in this subprocess was the fact that UiPath does not wait for the page to fully load to click the search result, for instance, whereas in Selenium it was needed a delay for the page to completely load and then click on the search result.

The second subprocess (Get files from SAS Visual Analytics), UiPath took less time than Selenium (3 minutes and 16 seconds for Selenium and 2 minutes and 6 seconds for UiPath). The difference between these execution times can be explained by the findings gathered in the previous section. First, in Selenium it was necessary to put delays (W3) to open the download toolbar and click on the download button (P6), whereas in UiPath it was able to click on a hidden download toolbar without opening the toolbar element (W7), saving time. Also, Selenium is not able to click on elements which are not present on the screen, so when it was needed to click on a tab that did not fit the screen (P7), it was necessary to click on a button to make the tab appear on the screen to be clickable (W5). To execute this subprocess, it was required to click on 11 tabs and download 11 files, so the extra steps plus the delays made Selenium take more time than UiPath.

Table 34 - Execution time of each subprocess with Selenium and UiPath in minutes/seconds

	Selenium				**UiPath**			
Subprocess	1	2	3	Total	1	2	3	Total
1#	*0:31*	*3:06*	*0:57*	*4:34*	*0:17*	*1:58*	*2:44*	*4:59*
2#	*0:33*	*3:36*	*1:05*	*5:14*	*0:18*	*1:57*	*2:45*	*5:00*
3#	*0:30*	*3:07*	*0:48*	*4:35*	*0:18*	*1:59*	*2:56*	*5:13*
4#	*0:35*	*3:12*	*1:08*	*4:55*	*0:19*	*1:56*	*2:44*	*4:59*
5#	*0:32*	*3:14*	*1:12*	*4:58*	*0:32*	*1:57*	*2:46*	*5:15*
6#	*0:34*	*3:14*	*1:12*	*5:00*	*0:19*	*2:00*	*2:45*	*5:04*
7#	*0:38*	*3:09*	*1:15*	*5:02*	*0:24*	*2:15*	*2:47*	*5:26*
8#	*0:37*	*3:11*	*1:10*	*4:58*	*0:20*	*2:01*	*2:44*	*5:05*
9#	*0:34*	*3:09*	*1:08*	*4:51*	*0:24*	*2:07*	*2:46*	*5:17*
10#	*0:34*	*3:16*	*1:12*	*5:22*	*0:22*	*2:01*	*2:49*	*5:12*
11#	*0:44*	*3:13*	*1:16*	*5:13*	*0:18*	*2:00*	*2:46*	*5:04*
12#	*0:35*	*3:19*	*1:06*	*5:00*	*0:18*	*1:53*	*2.44*	*4:55*
13#	*0:31*	*3:12*	*1:07*	*4:50*	*0:20*	*1:56*	*2:43*	*4:59*
14#	*0:35*	*3:07*	*1:11*	*4:53*	*0:22*	*2:08*	*2:46*	*5:16*
15#	*0:44*	*3:16*	*1:14*	*5:14*	*0:23*	*2:53*	*2:45*	*6:01*
16#	*0:44*	*3:06*	*1:12*	*5:02*	*0:21*	*2:07*	*2:44*	*5:12*
17#	*0:42*	*3:02*	*1:10*	*4:54*	*0:22*	*1:57*	*2:46*	*5:05*
18#	*0:42*	*3:02*	*1:12*	*4:56*	*0:33*	*1:53*	*2:47*	*5:13*
19#	*0:35*	*3:03*	*1:08*	*4:46*	*0:20*	*1:54*	*2:45*	*4:59*
20#	*0:29*	*3:31*	*1:05*	*5:05*	*0:19*	*1:53*	*2:46*	*4:58*
21#	*0:37*	*3:03*	*1:09*	*4:49*	*0:19*	*1:55*	*2:44*	*4:58*
22#	*0:40*	*3:07*	*1:13*	*5:00*	*0:23*	*2:06*	*2:46*	*5:15*
23#	*0:39*	*3:41*	*1:07*	*5:27*	*0:47*	*2:24*	*2:51*	*6:02*
24#	*0:40*	*3:19*	*1:20*	*5:19*	*0:24*	*2:10*	*2:47*	*5:21*
25#	*0:39*	*3:12*	*1:12*	*5:03*	*0:24*	*2:32*	*2:42*	*5:38*
26#	*0:39*	*3:11*	*1:19*	*5:09*	*0:29*	*2:28*	*2:46*	*5:43*
27#	*0:40*	*3:14*	*1:12*	*5:06*	*0:23*	*2:26*	*2:45*	*5:34*
28#	*0:45*	*4:01*	*1:17*	*6:03*	*0:30*	*2:02*	*2:55*	*5:27*
29#	*0:43*	*3:45*	*1:09*	*5:37*	*0:24*	*2:00*	*2:43*	*5:07*
30#	*0:38*	*3:13*	*1:09*	*5:00*	*0:23*	*1:57*	*2:43*	*5:03*
Average	*0:37*	*3:16*	*1:10*	*5:03*	*0:23*	*2:06*	*2:46*	*5:15*

On the last subprocess (Files upload to APM), Selenium took less time than UiPath (1 minute and 10 seconds for Selenium and 2 minutes and 46 seconds for UiPath) to execute. This is the subprocess that has the largest difference and the reason why Selenium has a lower execution time overall. There were many problems with clicking on the file selection input field with UiPath (P11), so large delays had to be inserted before clicking on the file selection upload field (W3), so that the file selection window would open, as described in the previous section. Also, Selenium was able to pass the file location into the file selection input field without opening the file selection window (W6). To execute this subprocess, it was required to upload 6 files, so the extra steps plus many delays made UiPath take more time than Selenium. If there were not any problems with the file selection input field, UiPath perhaps would take less time to execute all subprocesses.

After explaining the differences between execution times in Selenium Webdriver and UiPath, an explication of the technical differences based on the findings of automating the process is provided. When the robot is executed in Selenium Webdriver or UiPath, the interaction with the elements on the browser can be seen like a real person interacting with the elements. This visual way of visualizing the execution is useful for checking if the program works as intended and detecting bugs. In this process, one of the findings was that in Selenium in some tasks the robot was clicking on elements that were not clickable, which meant that it was performing tasks too fast. One example of that was hovering the toolbar and clicking right away on the download button to download a file on SAS Visual Analytics (P6). To solve the problem, it was necessary to add delays to click on the button only when the toolbar was loaded (W3). In UiPath, it was also necessary to add delays, for instance, before clicking on the file selection input field (P11) (APM website), so that the popup would completely load. The visual execution of the process in both Selenium and UiPath helped seeing that sometimes the execution runs too fast and delays are necessary so that no exceptions are thrown. In terms of interaction with elements, Selenium Webdriver is more restrictive, throwing exceptions easily, if for instance a button is clicked while it is disabled. The exceptions are useful because they provide more information about the error and combined with the visual execution of the process can help detect bugs and solve it easily. Plus, in Selenium Webdriver there are many conditions to wait until finding a element, such as "presence of element located", "visibility of element located", among others. Selenium Webdriver waits x seconds until the element is present in the HTML and then is able to click on it, for instance. UiPath was fewer waiting conditions to find the element (WaitForReady, WaitActivity and WaitVisible). These conditions are useful for elements that are not enabled or available on the screen right away, providing a way to wait until their state changes. UiPath, on the other hand, is more permissive in terms of the interacting with the elements. In the automated process, when clicking on the button to download a file on Service Now, the button is temporarily disabled and the UiPath robot clicks on it anyway without throwing exceptions (P10). This can be dangerous, because if the execution continues without performing one task, in the end the result of the execution can be unexpected and wrong. Moreover, as it does not throw an exception, it is more more difficult when a bug is detected to figure out why the program is not working.

Table 35 - Comparison between Selenium Webdriver and UiPath

Vector of analysis	Selenium Webdriver	UiPath
Goal	*Test automation*	*Robotic Process Automation*
Interaction with applications	*Web*	*Web and Desktop*
Type of software	*Open source*	*Proprietary*
Cost	*Free*	*Depends on the type of license*
Operating system	*All*	*Windows*
Software requirements	*-Python (or other language) installed* *-Download of driver for the chosen browser*	*None*
Programming skills	*Needed*	*Knowledge of logic*
Implementation time	*More*	*Less*
Implementation difficulty	*More*	*Less*
Execution time	*Less*	*More*
Maintainability	*Harder*	*Easier*
Technical	*Can see interaction with the browser*	*Can see interaction with the browser*
	More restrictive in terms of interaction with the elements (throws exceptions easily)	*More permissive in terms of interaction with the elements (e.g. clicking on a element that is disabled does not throw an exception)*
	Only able to click on elements that are visible on screen	*Can click on elements, even if they do not appear on screen (e.g. invisible elements)*
	Necessary to look at the HTML of the page to get the attributes for the element	*Can record interaction with the elements and get the attributes automatically – but sometimes does not pick the correct attributes for the elements*
	There are more waiting conditions to find the element: *-presence of element located* *-visibility of element located* *Etc*	*There are fewer waiting conditions to find the element:* *-WaitforReady, WaitActivity, WaitVisible*
	When the elements are on iframes, it is necessary to get the attribute that identifies the iframe and change into the context of the iframe	*Can identify elements, even if they are contained in iframes, without being necessary to change into the context of the iframe*
	Sometimes the execution runs too fast and it is necessary to insert delays into the program	*Sometimes the execution runs too fast and it is necessary to insert delays into the program*

Selenium Webdriver is only able to interact with elements that are visible on screen. If they are loaded but do not appear on screen, an exception is thrown. For instance, in the automated process when the robot wanted to click on a tab (SAS Visual Analytics) that did not appear on screen because of the size of the screen (P7), it was not possible and a button to make the tab appear on screen had to be clicked (W3). UiPath does not have that problem. In that case, UiPath was able to click on a tab that was not present on screen. Also, when it was needed to hover the toolbar and click to download a file (P6), UiPath did not need to open that toolbar and just clicked on a invisible download button (W7).

To get the attributes that uniquely identify the element, in Selenium Webdriver, it is necessary to look at the HTML of the webpage and find out manually what are the combination of attributes. This require some knowledge of how HTML works and it also time costly, because it trying to uniquely identify an element is a manually task. Another way, as explained in the previous section, is recording the interaction with the elements using Selenium IDE, get the attributes that identify the elements and copy their values to Selenium Webdriver. That approach does not always work, because Selenium IDE sometimes gets the wrong attributes, so looking at the HTML in such cases is required. In UiPath, it can record interaction and get the attributes automatically, but in some cases, it captures the wrong attributes. In those cases, there an UiPath option where one can select in a visual manner which attributes better identify the element, without having to look at the HTML.

In Selenium Webdriver, when the elements are inside iframe tags, it is necessary to identify the iframe element and change into the context of the iframe. Without knowing this, developers can think that elements are not being found because there have the wrong attributes and in a lot of cases it is because there are in the context of an iframe and in those cases it is necessary to change to the iframe context first in order to find the element. UiPath, on the other hand, does not have that problem. It can easily identify elements, whether they are inside an iframe or outside.

Chapter 8 – Develop Conclusions

RPA is being progressively used in companies and is revolutionizing the workplace by using software robots that mimic repetitive tasks that used to be done by workers, being faster and more efficient than humans.

As this type of automation becomes widespread, there are some issues, such as the cost of licensing. To reduce costs, this research aimed to find out if it is possible to use a test automation tool to automate BPs and identify the advantages and disadvantages of using a test automation tool as a RPA tool, compared to a RPA tool.

To accomplish that, this research automated a process with a test automation tool (Selenium) and with a RPA tool (UiPath) and compared both approaches, using a CS methodology.

This comparison was used to answer the defined research questions, which are answered as follows.

RQ1 – Is it feasible to use a test automation tool as a RPA tool?

It was confirmed that it is feasible to use a test automation tool like Selenium as a RPA tool. Despite some problems related in automating the chosen process in Selenium, such as being difficult to identify the elements and capture elements with attributes that are always changing, difficulties in finding elements inside a iframe tag and Selenium being unable to click on elements invisible on the screen, among others, many of the problems identified above were also reported in the chosen RPA tool (UiPath). This means that although there were problems in the automation, it is possible to automate a process in Selenium, but only websites can be automated as Selenium is only able to automate websites, meaning that if a process contains activities that require interacting with desktop applications, other technologies have to be used in order to automate that task.

RQ1.1 – What are the advantages and disadvantages of using a test automation tool as a RPA tool?

The advantages and disadvantages are represented in Table 36 and Table 37 (based on Table 35).

Using a test automation like Selenium can have multiple advantages but also disadvantages. As can be seen, there are more disadvantages than advantages, especially in technical aspects, such as only being able to click on elements that are visible on screen, the execution running too fast, the difficulty of identifying elements that are on iframes and the need to look at the HTML of the page to get the attributes for the element. Despite having more problems, some of them are common with UiPath, such as being difficult to identify the elements, among others.

Table 36 - Advantages of using a test automation tool as a RPA tool (RQ1.1)

Advantages
Open source
Can be used in all operating systems
Can see interaction with the browser
More restrictive in terms of interaction with the elements (throws exceptions easily)
More waiting conditions to find the element
Faster execution time

Table 37 - Disadvantages of using a test automation tool as a RPA tool (RQ1.1)

Disadvantages
Can only interact with web applications
Requires programming skills
More time to implement
More difficult to implement
More difficult to maintain
Only able to click on elements that are visible on screen
Necessary to look at the HTML of the page to get the attributes for the element
When the elements are on iframes, it is necessary to get the attribute that identifies the iframe and change into the context of the iframe
Sometimes the execution runs too fast and its necessary to insert delays into the program

Despite the disadvantages, automating processes with Selenium or other test automation tools can be useful in cases where it is necessary to automate processes that need to work in all operating systems and where only interacting with websites is required, as Selenium only interacts with websites.

These conclusions can help companies with low financial resources to find out if using an open source test automation tool is appropriate for their processes instead of using an expensive RPA tool, having into account the problems and advantages of using a test automation tool for RPA. Despite Selenium having no licensing cost, as it requires more development expertise, might need more manpower and maintenance. Therefore, it is necessary to develop a well-defined business case to make sure that it is an appropriate tool to use in that context. Also, this is an unexplored area and the results obtained can establish new baselines for future work.

8.1. Research Limitations

In this research, there are some limitations. As it only focuses on one process, the findings are only applicable to the chosen process and therefore cannot be generalized. If more processes were chosen and studied, more findings could be retrieved, and the conclusions would be more precise. Moreover, the answer to RQ1.1 was retrieved based on the comparison with UiPath and could be different if another RPA tool was used. Another limitation was that only one developer automated the process. If the development was performed by more developers, more opinions would be taken into consideration and the conclusions would be more rigorous.

8.2. Future Work

Future work should focus on applying Selenium or other test automation tools to automate more business process in different contexts and compare these test automation tools with other RPA tools, in order to strengthen and add more conclusions to the ones presented in this research. Plus, it can be interesting to explore the complementarity between RPA tools and test automation tools, taking advantage of their strengths and combining them to automate processes. Additionally, a more longitudinal study should be performed to compare the long-term effects of using a test automation tool as an RPA tool in terms of maintainability, as it can have a great impact in organizations.

Bibliography

Aalst, W. M. P. van der, Bichler, M., & Heinzl, A. (2018). Robotic Process Automation. *Business & Information Systems Engineering*, 1–4. https://doi.org/10.1007/s12599-018-0542-4

Aguirre, S., & Rodriguez, A. (2017). Automation of a Business Process Using Robotic Process Automation (RPA): A Case Study. *Applied Computer Sciences in Engineering*, 65–71. https://doi.org/10.1007/978-3-319-66963-2_7

Alberth, M., & Mattern, M. (2017, November). Understanding robotic process automation (RPA). *The CAPCO Institute Journal of Financial Transformation, Automation*(46). Retrieved from https://www.capco.com/-/media/CapcoMedia/Capco-Institute/Journal-46/JOURNAL46_full_web.ashx#page=104

Altaf, I., Dar, J. A., Rashid, F. u, & Rafiq, M. (2015). Survey on selenium tool in software testing. *2015 International Conference on Green Computing and Internet of Things (ICGCIoT)*, 1378–1383. https://doi.org/10.1109/ICGCIoT.2015.7380682

Anagnoste, S. (2017). Robotic Automation Process - The next major revolution in terms of back office operations improvement. *Proceedings of the International Conference on Business Excellence*, *11*. https://doi.org/10.1515/picbe-2017-0072

Anagnoste, S. (2018). Robotic Automation Process – The operating system for the digital enterprise. *Proceedings of the International Conference on Business Excellence*, *12*, 54–69. https://doi.org/10.2478/picbe-2018-0007

Asatiani, A., & Penttinen, E. (2016). Turning robotic process automation into commercial success – Case OpusCapita. *Journal of Information Technology Teaching Cases*, *6*(2), 67–74. https://doi.org/10.1057/jittc.2016.5

Bajaj, H. (2017). *Choosing the right automation tool* (p. 8). Infosys.

Baxter, P., & Jack, S. (2008). Qualitative case study methodology: Study design and implementation for novice researchers. *The Qualitative Report*, *13*(4), 544–559.

Cewe, C., Koch, D., & Mertens, R. (2017). Minimal Effort Requirements Engineering for Robotic Process Automation with Test Driven Development and Screen Recording. *Business Process Management Workshops*, 642–648. https://doi.org/10.1007/978-3-319-74030-0_51

Forrester. (2014). *Building A Center Of Expertise To Support Robotic Automation*. Retrieved from Forrester website: http://neoops.com/wp-content/uploads/2014/03/Forrester-RA-COE.pdf

Fung, H. P. (2014). *Criteria, Use Cases and Effects of Information Technology Process Automation (ITPA)* (SSRN Scholarly Paper No. ID 2540023). Retrieved from Social Science Research Network website: https://papers.ssrn.com/abstract=2540023

Geyer-Klingeberg, J., Nakladal, J., Baldauf, F., & Veit, F. (2018). Process Mining and Robotic Process Automation: A Perfect Match. *16th International Conference on Business Process Management 2018*, 9. Sydney.

Gupta, S., Kumar, S., & Saxena, C. (2015). *Review Paper on Comparison of Automation Testing Tools Selenium and QTP*. *5*(2), 3.

Hallikainen, P., Bekkhus, R., & Pan, S. L. (2018). How OpusCapita Used Internal RPA Capabilities to Offer Services to Clients. *MIS Quarterly Executive*, *17*(1).

Holmes, A., & Kellogg, M. (2006). Automating functional tests using Selenium. *AGILE 2006 (AGILE'06)*, 6 pp.-275. https://doi.org/10.1109/AGILE.2006.19

IEEE Guide for Terms and Concepts in Intelligent Process Automation. (2017). *IEEE Std 2755-2017*, 1–16. https://doi.org/10.1109/IEEESTD.2017.8070671

Islam, N. (2016). *A Comparative Study of Automated Software Testing Tools*. St. Cloud State University.

Jain, C. R., & Kaluri, R. (2015). Design of automation scripts execution application for selenium webdriver and test NG framework. *ARPN J Eng Appl Sci*, *10*, 2440–2445.

Juntunen, K. (2018). *Influence of contextual factors on the adoption process of Robotic process automation (RPA) : Case study at Stora Enso Finance Delivery* (Master's Thesis, Uppsala University). Retrieved from http://urn.kb.se/resolve?urn=urn:nbn:se:uu:diva-355079

Kämäräinen, T. (2018). *Managing Robotic Process Automation: Opportunities and Challenges Associated with a Federated Governance Model* (Master's Thesis, Aalto University). Retrieved from https://aaltodoc.aalto.fi:443/handle/123456789/32257

Kasslin, H. (2017). *Heavyweight and lightweight process automation - how do companies select between RPA and back-end automation?* (Master's Thesis, Aalto University). Retrieved from https://aaltodoc.aalto.fi:443/handle/123456789/26827

Kaur, H., & Gupta, D. G. (2013). *Comparative Study of Automated Testing Tools: Selenium, Quick Test Professional and Testcomplete*. *3*(5), 5.

Khramov, D. (2018). *Robotic and machine learning: How to help support to process customer tickets more effectively* (Bachelor's Thesis). Metropolia University of Applied Sciences.

Kirchmer, M. (2017). Robotic Process Automation–Pragmatic Solution or Dangerous Illusion? *BTOES Insights*.

Kitchenham, B. (2004). *Procedures for Performing Systematic Reviews* (p. 33). United Kingdom: Keele University.

Kyheröinen, T. (2018). *Implementation of Robotic Process Automation to a Target Process – a Case Study* (Master's Thesis, Aalto University). Retrieved from https://aaltodoc.aalto.fi:443/handle/123456789/31518

Lacity, M. C., & Willcocks, L. P. (2017). A new approach to automating services. *MIT Sloan Management Review*, *Fall*. Retrieved from http://sloanreview.mit.edu/

Lacity, M., & Willcocks, L. (2015). Robotic Process Automation: The Next Transformation Lever for Shared Services. *The Outsourcing Unit Working Research Paper Series*.

Lacity, M., Willcocks, L. P., & Craig, A. (2015). Robotic process automation: mature capabilities in the energy sector. *The Outsourcing Unit Working Research Paper Series*.

Lacity, M., Willcocks, L. P., & Craig, A. (2016). Robotic process automation at Telefonica O2. *MIS Quarterly Executive*, *15*(1).

Le Clair, C. (2018). *The Forrester Wave^{TM}: Robotic Process Automation, Q2 2018 The 15 Providers That Matter Most And How They Stack Up*.

Le Clair, C., Cullen, A., & King, M. (2017). *The Forrester Wave^{TM}: Robotic Process Automation, Q1 2017*. Retrieved from Forrester website: https://www.forrester.com/report/The+Forrester+Wave+Robotic+Process+Automation+Q1+2017/-/E-RES131182

Leopold, H., Aa, H. van der, & Reijers, H. A. (2018). Identifying Candidate Tasks for Robotic Process Automation in Textual Process Descriptions. *Enterprise, Business-Process and Information Systems Modeling*, 67–81. https://doi.org/10.1007/978-3-319-91704-7_5

Leotta, M., Clerissi, D., Ricca, F., & Spadaro, C. (2013). *Comparing the maintainability of selenium WebDriver test suites employing different locators: a case study*. 53. https://doi.org/10.1145/2489280.2489284

Lintukangas, A. (2017). *Improving indirect procurement process by utilizing robotic process automation* (Master's Thesis, LAPPEENRANTA UNIVERSITY OF TECHNOLOGY - School of Business and Management). Retrieved from https://core.ac.uk/download/pdf/84594991.pdf

Melchert, F., & Winter, R. (2004). Aligning Process Automation and Business Intelligence to Support Corporate Performance Management. *Proceedings of the Tenth Americas Conference on Information Systems*, 11.

Mindfields. (2015). *Robotic Process Automation Driving the next wave of cost rationalisation*. Retrieved from Mindfields website: https://www.scribd.com/doc/296828726/Robotics-Process-Automation-September-2015-v17-1

Moher, D., Liberati, A., Tetzlaff, J., & Altman, D. G. (2009). Preferred reporting items for systematic reviews and meta-analyses: the PRISMA statement. *International Journal of Surgery*, *6*(7), 336–341.

Patil, R., & Temkar, R. (2017). Intelligent Testing Tool: Selenium Web Driver. *International Research Journal of Engineering and Technology*.

Perry, D., Sim, S., & Easterbrook, S. (2004, June 23). *Case studies for software engineers*. *26*, 736–738. https://doi.org/10.1109/ICSE.2004.1317512

Ramya, P., Sindhura, V., & Sagar, P. V. (2017). Testing using selenium web driver. *2017 Second International Conference on Electrical, Computer and Communication Technologies (ICECCT)*, 1–7. https://doi.org/10.1109/ICECCT.2017.8117878

Raut, A. P., Chudiwale, G. A., & Kawale, P. A. (2016). A Survey and Study of Software Testing Tools. *International Journal of Research in Science & Engineering, Special Issue: Techno-Xtreme*, 212–217.

Robbins, J. (2012). *Learning Web Design, 4th Edition*. O'Reilly Media, Inc.

Runeson, P., & Höst, M. (2009). Guidelines for conducting and reporting case study research in software engineering. *Empirical Software Engineering*, *14*(2), 131–164. https://doi.org/10.1007/s10664-008-9102-8

Shaukat, K. (2015). *Taxonomy of Automated Software Testing Tools*.

Sigurðardóttir, G. L. (2018). *Robotic Process Automation: Dynamic Roadmap for Successful Implementation* (Master's Thesis, Reykjavík University). Retrieved from https://skemman.is/handle/1946/31385

Singla, S., & Kaur, H. (2014). Selenium Keyword Driven Automation Testing Framework. *IJARCSSE ISSN*, *2277*.

Slaby, J. R. (2012). *Robotic Automation Emerges as a Threat to Traditional Low-Cost Outsourcing*. Retrieved from https://www.horsesforsources.com/wp-content/uploads/2016/06/RS-1210_Robotic-automation-emerges-as-a-threat-060516.pdf

Stople, A., Steinsund, H., Iden, J., & Bygstad, B. (2017, November 27). *Lightweight IT and the IT function*. *25*. Retrieved from https://ojs.bibsys.no/index.php/Nokobit/article/view/405

Sualim, S. A., Yassin, N. M., & Mohamad, R. (2016). *Comparative Evaluation of Automated User Acceptance Testing Tool for Web Based Application*. *2*(2), 7.

Suri, V. K., Elia, M., & Hillegersberg, J. van. (2017). Software Bots - The Next Frontier for Shared Services and Functional Excellence. *Global Sourcing of Digital Services: Micro and Macro Perspectives*, 81–94. https://doi.org/10.1007/978-3-319-70305-3_5

Tellis, W. (1997). Information technology in a university: a case study. *Campus-Wide Information Systems*, *14*(3), 78–91.

Tellis, W. M. (1997). Application of a case study methodology. *The Qualitative Report*, *3*(3), 1–19.

Thomas, G. (2011). *How to Do Your Case Study: A Guide for Students and Researchers*. SAGE Publications.

Tornbohm, C., & Dunie, R. (2017). *Market Guide for Robotic Process Automation Software* (p. 35). Gartner.

Tran, D., & Ho Tran Minh, T. (2018). *Workflow Methodology Development of RPA Solution for A Vietnamese Bank: A Case Study of Korkia Oy* (Bachelor's Thesis, Laurea University of Applied Sciences). Retrieved from http://www.theseus.fi/handle/10024/148869

Unmesh, G. (2012). *Selenium Testing Tools Cookbook*. Packt Publishing.

Vila, E., Novakova, G., & Todorova, D. (2017). Automation Testing Framework for Web Applications with Selenium WebDriver: Opportunities and Threats. *Proceedings of the International Conference on Advances in Image Processing*, 144–150. https://doi.org/10.1145/3133264.3133300

Vishnu, S., Agochiya, V., & Palkar, R. (2017). Data-centered Dependencies and Opportunities for Robotics Process Automation in Banking. *Journal of Financial Transformation, 45*, 68–76.

Webster, J., & Watson, R. T. (2002). Analyzing the Past to Prepare for the Future: Writing a Literature Review. *MIS Q.*, *26*(2), xiii–xxiii.

Weske, M. (2012). *Business Process Management*. https://doi.org/10.1007/978-3-642-28616-2

Willcocks, L., Lacity, M., & Craig, A. (2017). Robotic process automation: strategic transformation lever for global business services? *Journal of Information Technology Teaching Cases*, *7*(1), 17–28. https://doi.org/10.1057/s41266-016-0016-9

Willcocks, L. P., Lacity, M., & Craig, A. (2015). The IT function and robotic process automation. *The Outsourcing Unit Working Research Paper Series*.

Yin, R. K. (1994). *Case study research: design and methods*. Sage Publications.

Yin, R. K. (2009). *Case Study Research: Design and Methods*. SAGE.

Zainal, Z. (2007). Case study as a research method. *Jurnal Kemanusiaan*, (9), 1–6.