

KARELIA UNIVERSITY OF APPLIED SCIENCES
Information Technology

Jesse Varis

**AUTOMATING PROCESSES IN WEB-INTERFACES WITH
ROBOTIC PROCESS AUTOMATION**

Thesis
December 2018



THESIS

NOVEMBER 2018

Information Technology

Tikkarinne 9

80200 JOENSUU

FINLAND

+ 358 13 260 600

Author (s)

Jesse Varis

Title

Automating processes in web-interfaces with Robotic Process Automation

Abstract

The primary goal of this thesis was to explore the automation of processes in web-based systems with Robotic Process Automation, a relatively new technology for the automation of repetitive and rules-based business processes. With the wide adoption of web-based systems and the recent surge of Robotic Process Automation in process automation, the combination of the two is a topical and an interesting subject to study.

This thesis comprises of two parts. The first focuses on the theory of Robotic Process Automation and the available solutions in general, as well as through the lens of web-applications. The second part defines use cases for Robotic Process Automation in two different web-applications and goes through the implementation as well as testing phases for the automation of these processes.

This thesis gives the reader a general idea of what Robotic Process Automation is, what it can be used for, and what it looks like in practice when applied to web-based processes.

Language

English

Pages 33

Keywords

Robotic Process Automation, UiPath

Table of contents

1	Introduction	6
2	Robotic Process Automation.....	7
2.1	Benefits.....	9
2.2	Capabilities and ideal processes	10
2.3	RPA in web-applications.....	11
3	RPA Tools.....	13
3.1	UiPath.....	14
3.1.1	UiPath Studio.....	15
3.1.2	UiPath Selectors.....	16
4	Use cases	17
4.1	Use case one: Moodle.....	18
4.1.1	Implementation and process flow	20
4.1.2	Problems and solutions	22
4.2	Use case two: Data scraping from YTJ.....	25
4.2.1	Implementation and process flow	26
4.2.2	Problems and solutions	27
5	Testing and results.....	28
5.1	Case one: Moodle.....	28
5.2	Case two: Data scraping from YTJ	29
6	Conclusion.....	31
	References.....	33

Abbreviations and terms

.Net Framework	Software development framework by Microsoft that is used to build applications that run on Windows Platform.
AI	Artificial Intelligence
API	Application Programming Interface, a definition of a set of rules or methods that allow software to access the features or data of an application.
CRM system	Customer-relation management system, a system used for storage and handling of customer data.
GUI	Graphical User Interface, the graphical component of a system, or a web page in the context of this thesis.
HTML	Hypertext Markup Language, the standard markup language for web pages.
Microsoft Workflow Foundation	A Microsoft technology that provides an API, an in-process engine, and a designer for the implementation of processes as workflows.
RPA	Robotic Process Automation
Robot	In the context of this thesis, Robot refers to the software that executes the automated processes.

VB.Net

Visual Basic .Net, a programming language on the .Net Framework.

XML

Extensible Markup Language, a markup language that is both human and machine readable.

1 Introduction

The primary goal of this thesis is to explore the current capabilities and possible challenges of automating processes in web interfaces using Robotic Process Automation. With the wide adoption of web-based systems, a substantial part of processes subject to automation involve manual work done in web interfaces. With that and the recent surge of Robotic Process Automation in the automation of business processes, this thesis aims to take a closer look at what automation in these systems involves, what the current capabilities of Robotic Process Automation tools are, and what problems may arise when making practical implementations.

A secondary objective for this thesis is to give the reader a general idea of what an RPA in practice is, and in a way that is easily readable and understandable. When researching other theses about RPA I noticed that there are not that many that focus on implementation on a more precise level. I feel that is a gap this thesis can somewhat fill. In addition to that, RPA is something that is heavily advertised as something not only for technical people, but for subject matter experts interested in automating their own processes too, so I think writing this thesis with that in mind is a good secondary goal.

This thesis comprises of two parts. The first focuses on the theory basis of RPA as well as on the current RPA products available. This part considers the elements of RPA in general, not only in web-applications. The second part is the implementation and testing of two use cases for RPA in web-applications.

The structure of the thesis is as follows. It begins in chapter two by defining RPA, its benefits, the ideal processes for it, and by exploring the use of RPA in web-interfaces from a theoretical point of view. In chapter three, the current key providers of RPA software are identified, and one is chosen for the practical part of the thesis. The chosen application and its features are then presented in the same chapter. Chapter four begins the practical part of the thesis by going

through the chosen cases for automation as well as their implementations. The testing and results for the cases are in chapter five. Finally, chapter six ends the thesis with conclusions.

It should be noted that as a fairly new industry, the literature of RPA is scarce and often not credible enough to be cited. As such, some statements lacking sources in this thesis are based on the experience and/or opinion of the author.

2 Robotic Process Automation

Robotic Process Automation, or RPA, is a relatively new technology for the automation of business processes with a core focus on automating repetitive, rules-based, and labor-intensive tasks. It consists of the creation and deployment of software robots that can mimic human behavior in the user interface of applications. This mimicked behavior can be things like data entry to forms or data extraction from documents. In practice, tasks are automated by producing a list of the actions performed in the task for the robot, and then letting the robot perform the task by running back that list of actions.

Previously it might not have been possible or worthwhile to automate or improve certain tasks because of constraints such as being stuck on old applications with lacking APIs or upgrading to new systems being too expensive. RPA solutions aim to make it possible to automate these tasks by modeling and configuring the task for the robot to perform. APIs, changes to software, or changes to the process itself are not required as the robot can perform the task in the graphical user interface in the same way a human would.

One of the primary goals of RPA solution providers is to make the process of building these robots as easy and fast as possible. That combined with the fact that almost nothing in the process itself needs to be changed is a big selling point for RPA as it lowers the barrier for entry on automating tasks and can lead to a high return on investment. Some of the products and advertisements for RPA go

as far as to avoid the association with software development altogether and advertise it as something anyone can do with very little training. That makes it tempting for companies to retrain their subject matter experts to automate the tasks in their own field. Whether retraining the workforce displaced by automation to build robots is a good idea or not remains to be seen though.

In the recent years, RPA has been growing extremely fast, with some of the solution providers raising hundreds of millions and reaching valuations in the billions in their funding rounds (UiPath 2018a & Automation Anywhere 2018). Not to mention that some of these providers have been in the business for only a handful of years. According to Everest Group (2018), as a whole the global RPA independent vendor market almost doubled to \$480-510 million in 2017, and a similar growth of 75-90% is expected for 2018. As for the adoption by businesses, recent and accurate numbers are hard to find, but in a 2017 study by Deloitte, 53% of 400 businesses they polled had begun their RPA journey, and that number is expected to rise to 72% by 2020 (Deloitte 2017). To get a rough idea of how new RPA and the interest in it is, figure 1 shows a graph from Google Trends that tracks the interest in it over time from 2012 to 2018. From the graph, the start of the quick rise of RPA can be traced to somewhere in the beginning of 2015.

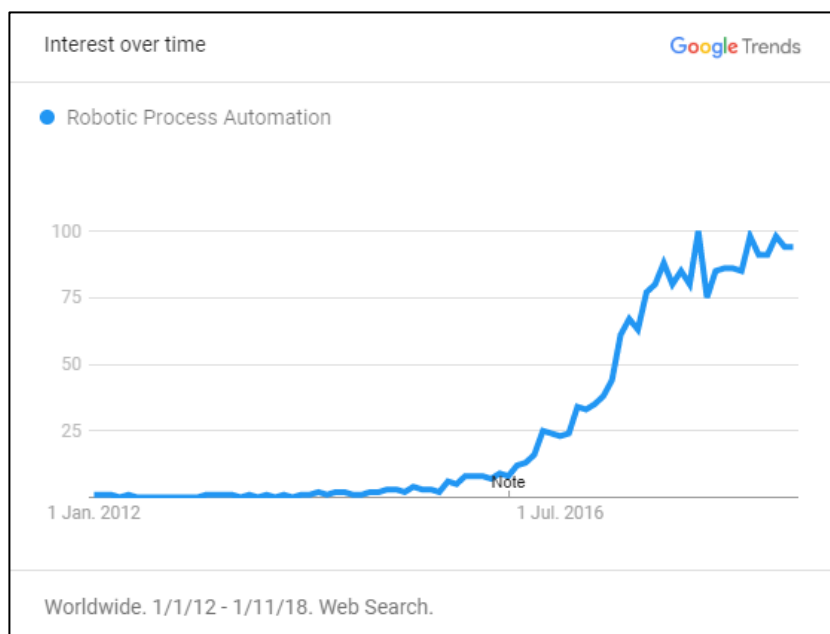


Figure 1. Interest over time in “Robotic Process Automation”, from Google Trends

As for the technology itself, it is perhaps not as revolutionary as one might think, since many of its features existed in automated testing software and screen scraping technologies before RPA as a term was ever conceived (Arrow Digital 2018). What distinguishes RPA from its humble beginnings though is that it packages all of that and much more into one solution that can automate in practically any application.

Also, one slightly misled association that is often made is with RPA and intelligent automation. On its own, RPA is not intelligent. RPA requires the process and its rules to be completely defined and configured for the robot. The robots themselves do not, for example, have the ability to learn. In the future though, RPA will likely be combined with machine learning and AI solutions to automate more complex processes. Some cases for this already exist, e.g. Hollard Group combined RPA, machine learning, natural language processing, optical character recognition and analytics to process their 1.5 million yearly insurance claims (UiPath 2018i).

2.1 Benefits

The most obvious benefit of RPA is the monetary benefit of reducing labor costs by moving tasks from humans to robots. The annual cost of a single robot in 2017 was from \$5000 to \$10000, which for the amount of work it can perform compared to a human is relatively cheap (Forrester 2017). For optimal processes, the potential return on investment is clearly huge. The price of a robot does not consider other costs though, such as infrastructure, development or maintenance. The robots need their own environments to run in, and someone must build them as well as maintain them in case they break or the process changes. For smaller companies, the costs could quickly catch up with the savings gained through reducing labor costs, making some processes not qualify for automation from a purely financial perspective.

Luckily, saving money is not the only benefit. Being software, the robots are much faster than humans and they can work around the clock. This means they can complete tasks in less time and whenever desired. Furthermore, they do not make mistakes like a human would, because they perform the task exactly the way they are configured to. All of this can massively improve the aspects of quality and speed of completion in the process. For example, in case of a time sensitive process the robot can run it during the night and have it completed before a human would have even started working on it in the following morning. (AppliedAi, 2018b)

As all their actions can be logged and traced when needed, the robots are also easier to monitor than humans. This can improve security and compliance. In case of an error, the exact actions of a robot can be traced, and the location of the problem found instantly, as opposed to trying to figure out what a human did and when. Logs can also potentially provide higher quality analytics for the processes, giving more insights in the overall quality of the process and the parts of it that need further improvement. (AppliedAi, 2018b)

2.2 Capabilities and ideal processes

In the UI, robots should be able to perform practically any action that a human can. Robots can navigate in systems, read structured data from interfaces, read or write files, and so on. In addition to that, as it is software it can also do things such as connect to APIs, read and write to databases, or execute complex calculations.

What RPA on its own cannot do is essentially anything that requires human judgement. Things like reading from unstructured data or improvising in case of unknown exceptions. Unstructured data in practice can be, for example, a text field with instructions written by a human that needs to be interpreted in order to know how to continue in a process.

As for ideal processes, they are the ones that can be clearly defined down to a set of rules, that have a lot of repetition, and that currently consume a lot of manual human labor. Processes fulfilling those three requirements will most likely lead to the biggest return on investment. It also helps if the process is as straightforward as possible, since defining exceptions for a robot can get laborious and therefore expensive. The process and the systems involved in it should also be stable since changes to them also require changes to the robot, which will in turn make the maintenance expensive. These are perhaps the most important ones, but there are many other aspects to consider, such as the strategic value of the process to the company, employee or customer satisfaction gained from its automation, potential quality improvements, and so on. (AppliedAi, 2018b)

2.3 RPA in web-applications

Like in any other application, RPA can perform in web-applications basically any straightforward actions that a human can. To do that, it needs to be able to identify the elements that it wants to interact with. For that, there are practically two techniques. One is using image capturing and screen position, and the other is identifying the element from the underlying structure of the application.

Image capturing in this context means using a previously saved image to identify the position of an element on the screen and it is the inferior way as it only considers the 'where' (on the screen) part of the equation. Knowing only the position on the screen, actions are limited and e.g. scraping the text must be done by manually copying it to the clipboard. When using image capturing, scraping the text can become impossible in some cases as the elements can have text that cannot be selected or the elements themselves can be completely hidden from the view.

Much more can be done when identifying the individual elements straight from the application structure, as the elements and their attributes can be directly interacted with. For example, a text within them can be read or modified

programmatically. With web applications, accessing the underlying structure is not a problem since they are opened and rendered in the browser. The standard markup language that browsers read and render web pages from is HTML. To put it simply, HTML consists of nested elements and their attributes. As HTML is the standard and standardized, RPA solutions can use its attributes, among other things, to uniquely and reliably identify elements in web-applications.

There are some things that could make web automation difficult though. Obviously, web pages are not built with RPA in mind, and the underlying structure can be a mess even if the page looks normal on the surface. The structure being a mess or the page having been built badly can make identifying the elements harder, or cause other unreliability issues. On the other hand, the page having been built well does not mean that it benefits RPA either. A lot of modern pages have e.g. dynamicity on the surface of the page as well as under it. On the surface, things constantly moving or changing can complicate actions like clicking on the elements. Under the surface, the elements or even whole segments of the page can be dynamically created, meaning that all their attributes can be dynamically generated as well. The elements being dynamic could make uniquely identifying them harder, or in some cases impossible.

Some other things to consider with web-applications are updates and connection reliability. With locally installed applications, you can control or at least know its update schedule, and as it is locally installed, connecting to it is not a problem. With web-applications, neither of these is guaranteed. In a worst-case scenario, the update and maintenance schedule of the web-application used is completely unknown as well as unreliable. Updates can cause huge problems as changes to the page can change the elements or their attributes, which can break the automation. The page unexpectedly going on a service break when the robot is supposed to run will lead to an error and increased maintenance costs of the robot as someone needs to check these errors. As for the connection, when relying on connecting through the internet, it can be slow or not work at all, on their end or on your end, which again can lead to errors.

With all that, the simpler the page the better. One could even say that the older the page, the better. Old and simple web-pages do not usually have all the bells and whistles of a modern web development causing issues and are instead static as well as somewhat reliable. They are also more likely to lack the fancy integration tools and APIs that would probably be utilized before resorting to RPA.

3 RPA Tools

There is quite a lot to choose from when it comes to RPA tools, as the offering has grown fast along the business in the last few years. In June 2018, Forrester counted 32 different product vendors in their report on the current RPA solution providers (Forrester 2018). In addition to those with paid licensing models, free and open source solutions have also started to emerge. For example, Robot Framework, a Finnish open source test automation framework that expanded into RPA in 2018 (Siili Solutions 2018).

In their report, comparing the RPA product offering, Forrester analyzed and scored 15 of the 32 providers they deemed most significant, evaluating them on 30 different criteria. Based on their current offering and strategy, Forrester chose UiPath, Automation Anywhere, and Blue Prism as the three leaders in the field. The only one that came even close was WorkFusion, a solution that focuses on AI. The results did not come as a surprise as UiPath, Automation Anywhere, and Blue Prism are the ones that come up first in almost every context when researching RPA. Out of the three, Forrester ultimately gave UiPath the best score, while also giving it by far the best score in many of the technical aspects. (Forrester 2018)

Besides the technical aspects, UiPath also demonstrates the most potential out of the three leading RPA products. As can be seen from Figure 2, UiPath is currently leading the pack, but a year earlier in the first quarter of 2017 when Forrester made a similar report, UiPath was placed third with especially weak market presence compared to the other two leaders. So, within a year it has

passed or at least caught up with the other two on seemingly all aspects. (Forrester 2017)

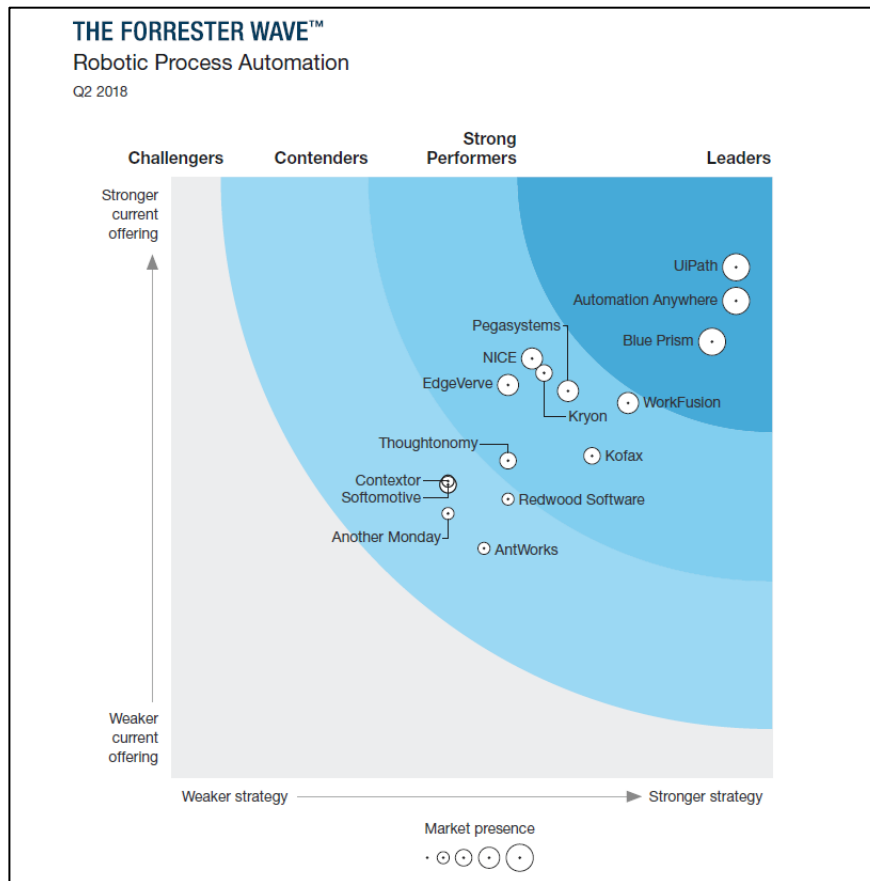


Figure 2. Forrester's 2018 comparison of RPA products

As the focus of this thesis is on exploring the current capabilities of RPA, I am obviously picking the best solution for the practical part, or UiPath in this case. UiPath also happens to offer the most extensive trial version out of the three, with Automation Anywhere only having a 30-day trial, and Blue Prism not offering one at all.

3.1 UiPath

UiPath, founded in 2005, is a software company originating from Romania that specializes in RPA. They launched their first RPA product in 2013, after which their business has been growing exponentially (UiPath 2018d). In the beginning of 2018, they reported a growth in enterprise customers from one hundred to

seven hundred during the year 2017 (UiPath 2018a). After the first half of 2018, that seven hundred had grown to over one thousand five hundred (UiPath 2018b). In 2018, the company also reached a valuation of over \$1,1 billion in their Series B funding, where they raised \$153 million (UiPath 2018a). And since that was not enough, during the writing of this thesis UiPath held another funding round, raising \$225 million for a valuation of \$2,9 billion (TechCrunch 2018).

Their flagship product is called UiPath platform. It consists of three components, those being UiPath Studio, UiPath Robot, and UiPath Orchestrator. UiPath Studio is the development environment where the automation workflows are built, UiPath Robot is used to run the automations, and UiPath Orchestrator is used for scheduling and monitoring the robots as well as providing analytics of them. (UiPath 2018f & UiPath 2018g & UiPath 2018e)

UiPath offers a free trial version of their product, called UiPath Community Edition. UiPath Community Edition includes all the functionalities of UiPath Studio and UiPath Robot needed to build and execute robots. What it does not include is local hosting of the Orchestrator, and instead offers an online version for testing purposes. Orchestrator, however, is not needed in the scope of this thesis. (UiPath, 2018c)

3.1.1 UiPath Studio

UiPath Studio is a visual development environment in which you build your automations. It is built upon the Microsoft Workflow Foundation of the .Net Framework and is based on the idea of building workflows made of sequential programming steps called activities (Microsoft 2018). An activity is a step in the workflow that in UiPath can be anything from a click to sending an email or creating a file. All the programming functionality is contained within these activities, for the purpose of keeping the workflows visual and easy to follow. Hundreds of core activities are included in the UiPath Studio.

The building of workflows in the studio is done by dragging and dropping activities from the sidebar and into the workflow. There are three types of workflows; sequences, flow charts and state machines. Multiple activities are always contained in a sequence, which is just a linear container for activities. An example of a sequence containing activities can be seen in Figure 3. Activities and sequences containing activities can then be used in flowcharts or state machines for a more complex workflow logic.

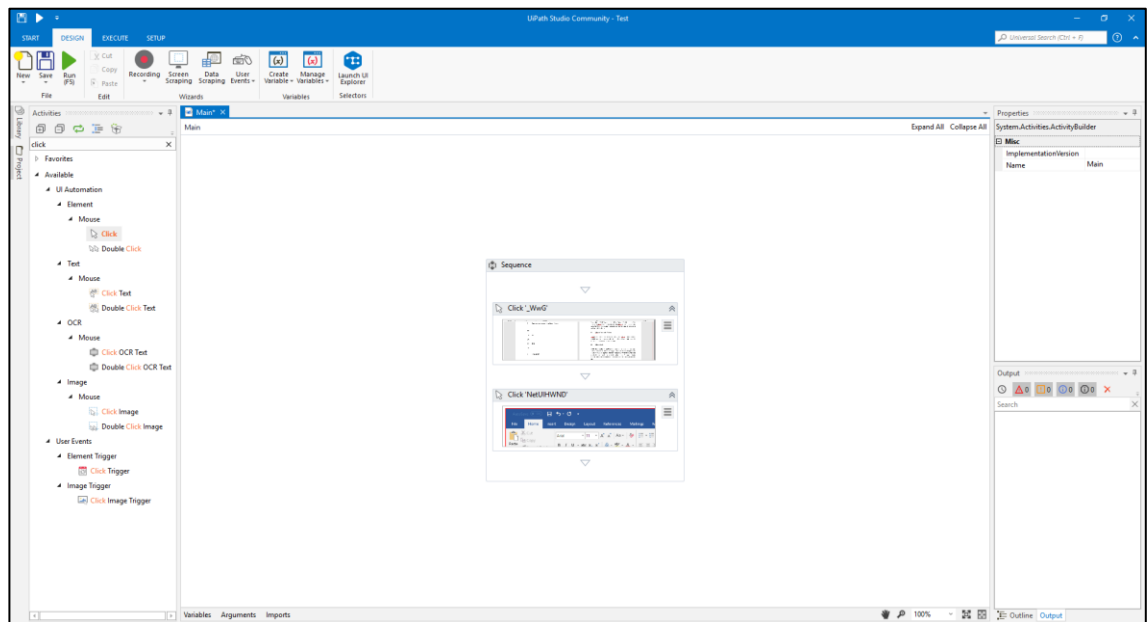


Figure 3. UiPath Studio, with a sequence containing two click activities.

On the surface, UiPath Studio looks code free as the programming functionality is contained within the activities. However, since it is built on the .Net Framework, it allows the use of all VB.Net variable types in addition to their own, as well as the usage of .Net functions when working with these variables within the activities. The building and usage of custom-made activities as well as invoking of VB.Net code in the workflow is also supported, in case the execution of a more complex code is needed. All that allows practically as much depth as needed while keeping the surface simple.

3.1.2 UiPath Selectors

What UiPath claims is the secret to their resilient core is “the selector”. Selectors are a technology which UiPath developed years before entering the RPA market (UiPath 2018e). A selector is a string of XML that UiPath activities use to identify elements in applications. Basically, they are a collection of the attributes of an element and its parents in the UI that can be used to uniquely identify that element, so that actions like clicks can be performed on it. The selector technology works on any application in the Windows ecosystem, not just web applications. UiPath automatically generates selectors for activities, but they can also be made manually and customized for more accuracy and robustness. An example of a selector pointing to the search field in google.com can be seen in Figure 4.

```
<html app='chrome.exe' title='Google' />  
<webctrl name='q' tag='INPUT' />
```

Figure 4: An example of a selector, pointing to the search field in google.com.

For selector customization, UiPath has a tool called UiExplorer that comes with the studio. In it the elements can be inspected, and their selectors customized. It also provides a visual tree of the application document object model, where the elements can be searched and selected for selector creation. It is a useful tool as the alternative would be inspecting the elements from the browser.

4 Use cases

When choosing the processes for the practical part of this thesis I wanted to have ones that would demonstrate different use cases for RPA, as well as ones that would be easy to understand and compare to real-world use cases. I also wanted them to display the distinct task types for robots, those being data scraping and data input.

For the implementation, there were a couple of options. Mainly, there was the option of using UiPath's own project template called UiPath ReFramework, which is provided with the studio. It is built on top of the transactional business process template and uses the state machine layout for top level process flow. It also has a couple of useful built-in things like error handling, project initialization and configuration files, and credential fetching. With all that, it could have been used for both the cases presented in this thesis.

However, to keep things simple, I did not use the template and opted to build the projects from scratch instead. I did use the state machine layout, as it seems to be what UiPath themselves prefer, and does offer some more flexibility than a flowchart, which is the other option. How the state machine works in UiPath is that there are states and transitions. The states have entry and exit blocks where activities can be executed, and transitions are essentially movements between the states. Multiple transitions can go in and out of states and their condition is used to decide which one to follow. Activities can also be inserted within the transitions if needed.

As for exception handling, it was done similarly to how it is done in the ReFramework template, as in by using UiPath's try-catch activity within the states to encapsulate all the other activities and marking the error in a variable in the catch block when one occurs. Transitions then check for that variable and move to an appropriate state, e.g. the end step of the process in case of an error.

4.1 Use case one: Moodle

The first use case is for the free and open source learning management system Moodle. Moodle is widely used all around the world, claiming 138 million users globally (Moodle 2018). That hopefully makes it a relatable and an easy to understand example for most people. It also offers an easy local installation which is ideal for a thesis as the use isn't restricted in any way.

As for the process itself, I chose course creation as it is a simple but an easily expandable example. A real-world case could be one where there is a large number of courses in another learning management system that need to be moved to Moodle, and straight up export and import is not supported. This is where a robot could be configured to perform the manual labor of extracting the courses, perhaps performing some parsing, and then creating the courses in Moodle. At bare minimum it could involve creating the courses with the basic information, or it could involve moving the course content, files, etc. as well.

For this thesis, I focused exclusively on the course creation part of that imaginary process, as I mainly wanted this case to demonstrate data entry. All the needed information for creating the courses is already provided for the robot in an Excel file. The Excel file has a table format where courses are in rows. The row contains basic information and topic names as well as topic content for four topics. An example of the Excel is in Figure 5.

	A	B	C	D	E	F	G	H	I	J	K
1	Full name	Short name	Summary	Topic 1 name	Topic 1 content	Topic 2 name	Topic 2 content	Topic 3 name	Topic 3 content	Topic 4 name	Topic 4 content
2	United States	USA	The United	History	The first	Population	The U.S. Census	Language	English (American	Economy	The United
3	Finland	FIN	Finland ,	Etymology	The earliest	History	If	Geography	Lying	Politics	The Constitution

Figure 5. An example of the Excel used in the first case

With that, the process becomes a relatively simple one where the robot first reads the Excel file, logs in to Moodle, and then creates a course for each row provided in the Excel. A step by step representation of the process contains the following steps:

1. Read the Excel file
2. Open and log in to Moodle
 - 2.1. Navigate to site home page
 - 2.2. Turn editing mode on
3. For each row in the Excel file
 - 3.1. Click “Add new course”
 - 3.2. Fill in the basic information (name, short name, summary)
 - 3.3. Click “Save and display”

- 3.4. Click “Proceed to course content”
- 3.5. For each topic
 - 3.5.1. Change the topic name
 - 3.5.2. Open the topic content editor
 - 3.5.3. Add the topic content
 - 3.5.4. Save and return to course main page
- 3.6. Return to the site home page
4. Close Moodle

4.1.1 Implementation and process flow

To start with, the complete top-level process flow in a state machine can be seen in Figure 6. The states are somewhat like the highest level in the step-by-step description of the previous chapter. First, there is the “Init” state where the robot reads the Excel file into a datatable variable within UiPath. Then there is the “Login state” where it logs into Moodle, and the “Read Row” state where it reads the next row in the datatable. If a row exists, the “Create course” state is executed, and the robot creates the course in Moodle. If the creation is without errors, it moves back to the “Read row” state. When the “Read row” state no longer finds a new row, the robot transitions to the “End” state where the browser is closed and the execution ends.

Error handling is done with the transitions that have the “Error” text as well as the “Out of rows” one. In all of them except the one after the “Create course” state, in case of an error the transition leads to the “End” state. If an error occurs in the “Create course” state, the transition leads to the “Login” state where the robot logs in to Moodle again and then continues the execution from the next row. In a real-world scenario, the robot would probably also mark down the courses that it failed to create, as well as other errors, and report those e.g. by email.

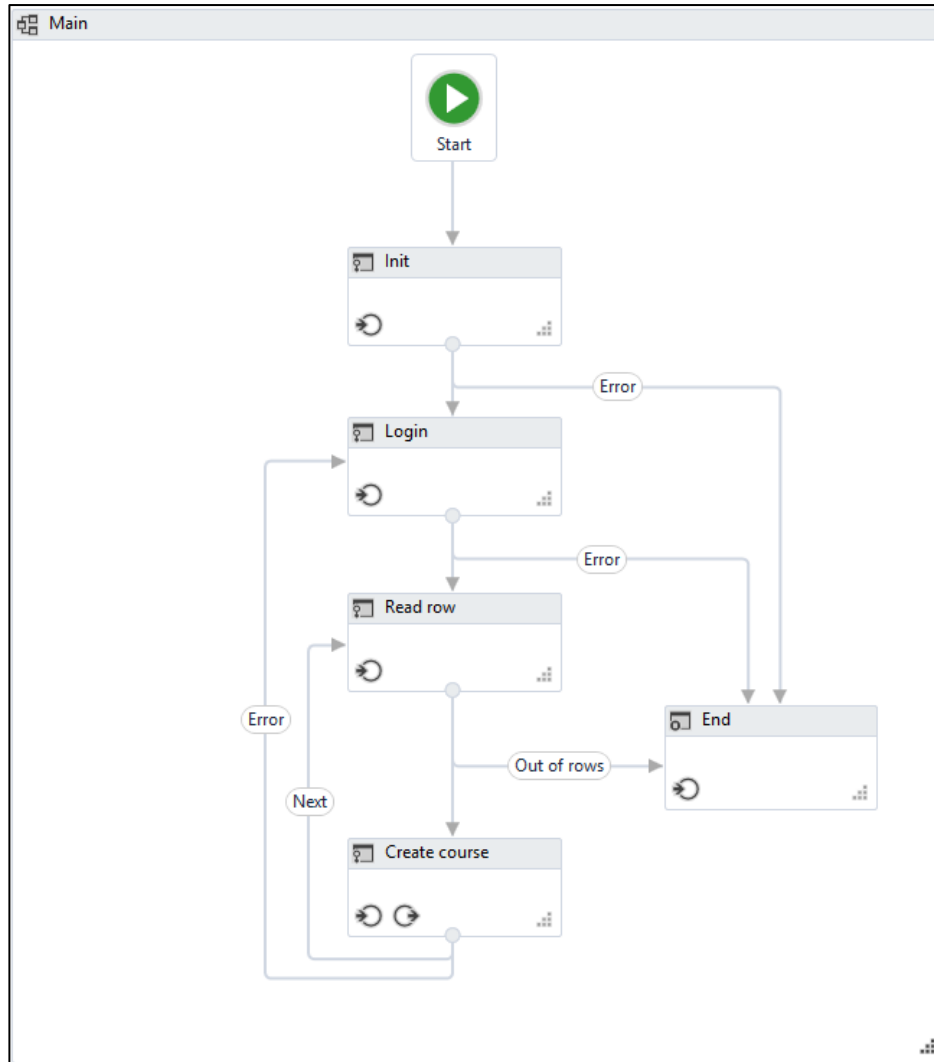


Figure 6. Top-level process flow in a state machine for the first case

The activities that execute the actual actions required in each state, such as clicks and typing text in the forms, are all contained within the states. I will not go in detail about them, but a sample of what they look like in the “Create course” state is provided in Figure 7.

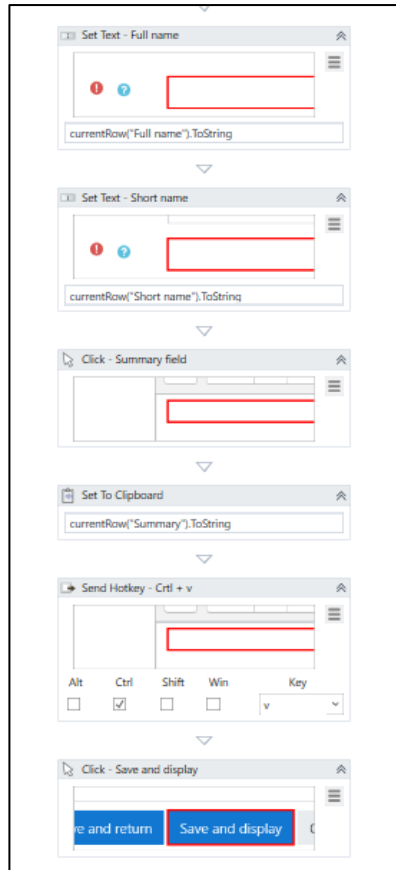


Figure 7. Sample of the activities in the “Create Course” state

4.1.2 Problems and solutions

Even though Moodle is and looks fairly simple, there were multiple small things that required alternate solutions to the obvious one, and ones that I noticed could be unreliable in the long run. There were also some issues with the robot being too fast in its execution.

Most notably, while adding the content to the course, there were a couple of spots where completely reliable selectors could not be created, and less reliable ones had to be used instead. This was mostly because Moodle creates four empty topics to new courses by default, and the way they are built is very similar to each other. For example, when adding content to the topics, the edit dropdown for that topic section needs to be clicked before opening the page with the content editor, and the way dropdowns are identified by Moodle seems

to be dynamic. An example of this with the edit dropdowns pointed out by purple squares and their selectors added next to them can be seen in Figure 8.

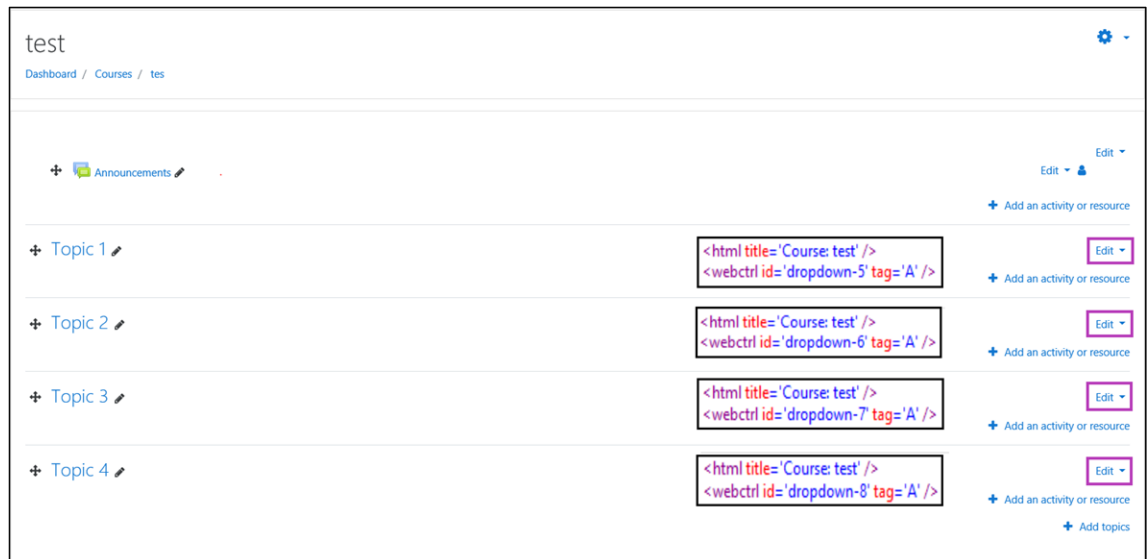


Figure 8. A new Moodle course

The problem is not that the selectors do not work, but that the identifying part of the element, or id in this case, seems to be dynamically created. For each dropdown, Moodle appears to use a similar id, but with an incrementing index at the end. If that is the case, the selectors for these elements only work for as long as new dropdowns do not appear on the page before the ones used, and when they do the robot will either add the content on the wrong sections or fail as it will try to open a section that does not exist.

As for how one could make a more robust solution in this situation, the search from the page could be limited to the topic section that needs to be edited, so only one dropdown is found. That can be done in UiPath with an activity called Find Children. Find Children can be used to find the children or descendants of a specified element based on a filter. With the topic section as the specified element and the dropdown attributes in the filter, it only returns one element, the dropdown element for that section, and that can then be used in the click activity that opens the dropdown.

One smaller but slightly interesting thing was that the text could not be straight up inserted into the text editor fields of Moodle (example in Figure 9). In this process, there were text editor fields like that when adding the course summary and topic contents. Basically, there are two ways of inserting text into fields in UiPath, either with the “Set text” or “Type into” activities. “Set text” is instant as it inserts the text into the text attribute of the element, and “Type into” types the keystrokes as a human would. The text editor is probably built in a way that it does not have the text attribute that “Set text” uses, so it does not work. “Type into” does work but is in this case unusable as the course summary can be so long that the robot will type the text into it for an unreasonably long time.

The most practical solution in this case was using hotkeys, or more precisely copying the text into the clipboard and pasting it into the text field using the ctrl+v hotkey. For both, UiPath has activities by default, so no VB.net code is needed.

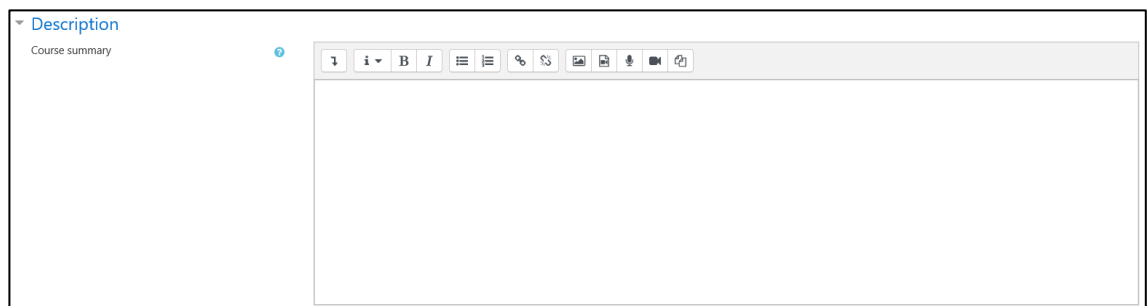


Figure 9. A text editor field in Moodle

Another even less significant thing was that when adding the topic names, it seemed impossible to inspect the element where the topic name is written using UiPath’s features. The selector that UiPath automatically created was not suitable so it had to be edited, but as the field where the text is written into disappears on basically any action that is not typing into it, UiExplorer was not able to inspect it. In this case, I had to create the selector “manually”, as in by inspecting the element in the browser and getting the attributes from there.

Apart from these things, there were not really any problems with the implementation, and only a couple of things needed to be modified before moving on to testing.

4.2 Use case two: Data scraping from YTJ

For the second process, I wanted a typical data scraping process. One where data is searched, scraped and stored or sent somewhere. An example of a real-world case similar to the process I am going to use here would be extracting customer information from a CRM system for the purpose of moving it elsewhere. This could be necessary, for example, when updating an old CRM system to a new one, and if the old one did not provide APIs or other features for the extraction of the data.

Data scraping itself is a very broad term and there are an infinite number of different things you could extract from the web. The one I chose, though, is scraping basic company information from YTJ, the Finnish business information system. As an input, the robot will have a csv file with a list of business IDs. It will perform a search on the business IDs on the YTJ website and scrape the predetermined set of information for each company. The scraped information will be stored in an Excel file. Step by step, the process will be as follows.

1. Read the csv file
2. Open “<https://tietopalvelu.ytj.fi/>” in the browser
3. For each row in the csv file
 - 3.1. Search for the company
 - 3.2. Select and open the company from search results
 - 3.3. Scrape the following information
 - 3.3.1. Trade name
 - 3.3.2. Company form
 - 3.3.3. Home municipality
 - 3.3.4. Line of business
 - 3.3.5. Address

- 3.3.6. Phone
- 3.4. Write the scraped information into the Excel
- 3.5. Return to the search page
- 4. Close the browser

4.2.1 Implementation and process flow

As can be seen from Figure 10, the top-level process for the second process ended up looking almost identical to the first one. This is mostly because they both follow the same logic of looping rows of data and performing an action on that data, only in this one instead of using that data to create something it is used as a foundation to search, scrape and store other data. I also used the state machine from the first one as a template for this one, which contributed to them looking similar.

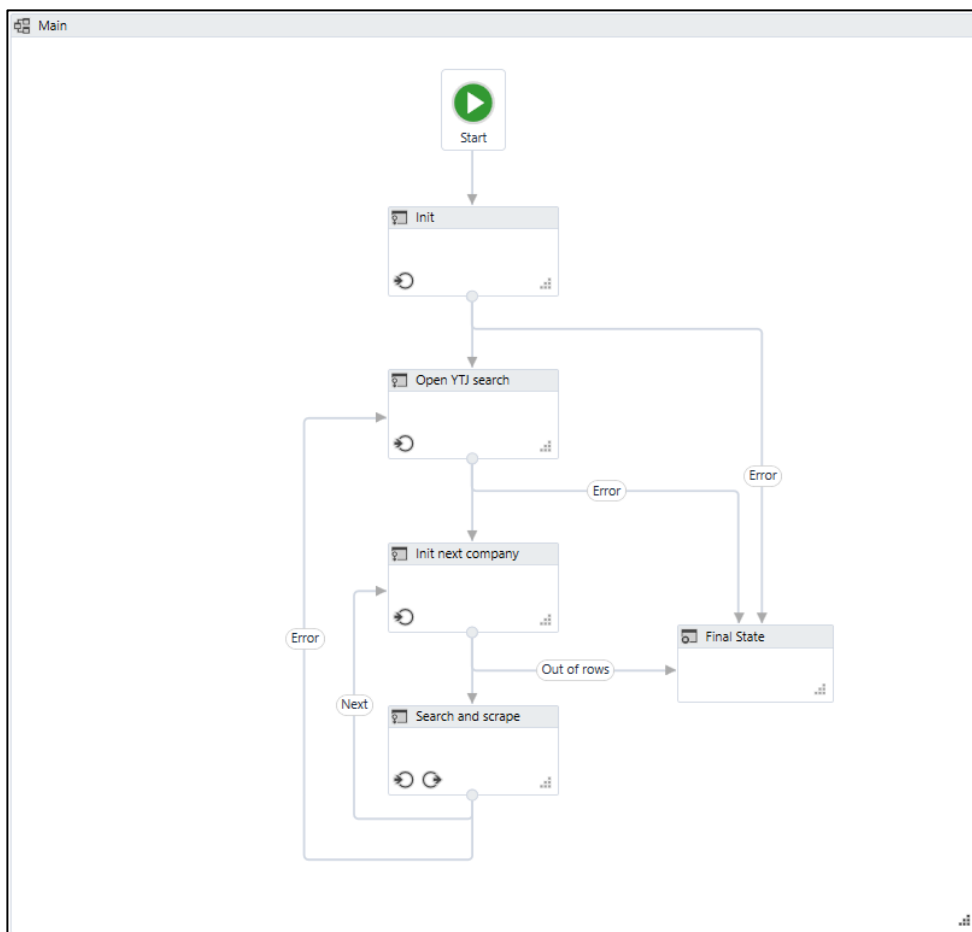


Figure 10. Top-level process flow in a state machine for the second case

Arguably, the process could have been separated into more states in this case, e.g. the saving of the data into the Excel file happens in the same state as it is scraped in and could have been separated into its own. But as the process is small and the saving and scraping do not warrant separate error handling, I kept it them in the same one.


Like the state machine, the process flow for this implementation is very similar to the first case. In the “Init” state, the robot, in addition to reading the input file with the business IDs, also creates a datatable template where the scraped information will be stored before writing it into the Excel. It then opens the YTJ business search page in the “Open YTJ search” state, checks for the next company on the input list in the “Init next company” state, and searches for the company, scrapes the data and saves the data for that company in an Excel file in the “Search and scrape state”.

4.2.2 Problems and solutions

This implementation ended up being more straightforward than the first one, as there were not really any significant surprises with the search form, its results, or their underlying format. I did note that the table elements on the page did not have identifying attributes, which could lead to possible mix-ups between tables were they to be too similar in content. For the data collected here though, the rows and columns within the tables were named though and the table structure itself was normal, so it did not cause any issues. Apart from that note and the fact that there appeared some whitespace in some element attributes which had to be fixed with wildcards, everything went smoothly.

However, one thing I would like to elaborate on about the table structure is that even though it did not end up affecting this implementation, I noticed that there was dynamicity within some of the tables. For reference, there is an image of the page containing the information in Figure 11. The dynamicity combined with the fact that the tables themselves were not uniquely identifiable apart from their

position on the page, could in some cases lead to the robot reading information from the wrong table. As an example, if reading from a table that only has identified columns and not rows, if another table with the same column name appeared, the two could get mixed up by the robot. But again, this was not an issue in this implementation, just something I took note of.



[Suomeksi](#) | [På svenska](#)
[Log in with your Katso ID](#)

KONE Oyj [Print Page](#)

Go to the [Virre website](#) of the National Board of Patents and Registration to perform more precise Searches of Trade Register information and the Register of Foundations.

Business ID: 1927400-1

	Latest information	As of	Information source
Trade name	KONE Oyj	06/01/2005	Finnish Patent and Registration Office
Parallel trade name	KONE Corporation	06/01/2005	Finnish Patent and Registration Office
Auxiliary trade name	Kone Engineering	06/01/2005	Finnish Patent and Registration Office
Company form	Public limited company	06/01/2005	Finnish Patent and Registration Office
Home municipality	HELSINKI	06/01/2005	Finnish Patent and Registration Office
Language	Finnish	06/02/2005	Tax Administration
Main line of business	Manufacture of lifting and handling equipment (28220)	12/31/2007	Tax Administration
Postal address	PL 8 00331 HELSINKI	11/01/2004	Common
Street address	Kartanonlie 1 00330 HELSINKI	11/01/2004	Common

Registrations in force

Register	Status	As of
Trade register	Registered	06/01/2005
Tax Administration	Registered	06/02/2005
Prepayment register	Registered	06/01/2005
Value added tax-liability	VAT-liable for business activity	06/01/2005
Value added tax-liability	VAT-obliged for the transfer of rights to use immovable property	06/01/2005
Employer register	Registered	06/01/2005

[Show registration history](#)

Prepayment register check dates

in 2018	in 2019 and after
28 February, 1 September and 1 December	1 March, 1 June, 1 September and 1 December

Tax Debt Register

[Show details](#)

Business ID history

Date	Event	Note
11/01/2004	ID given	
06/01/2005	Division	Business 0110139-9 has demerged into the following company: 1927400-1.
02/28/2007	Fusion	Business 1902018-3 merged into 1927400-1.
09/30/2009	Fusion	Business 1017435-4 merged into 1927400-1.

Figure 11. An example of the page from which the data is scraped

5 Testing and results

5.1 Case one: Moodle

After being content with the implementation, I ran the process through a few testing rounds. For the testing I created five different course templates with varying amounts of content.

For the first test I ran the process with 20 courses, creating each course four times. The execution took 12 minutes and 44 seconds, or around 38 seconds per course. Out of 20, the process failed two times with both happening in the same part of the process and for the same template. The part that failed was opening the topic content editor, or specifically opening it after returning from another content editor.

What happened was that the robot missclicked the edit dropdown button, because after returning from another content editor, Moodle places the beginning of that topic on the page to the top of the browser, and this happens with a small delay. With the delay, the robot sometimes tried to click on the next edit dropdown at the exact same time that the page was shifting, causing the click to miss.

One way to prevent these kinds of errors in UiPath is the `WaitForReady` property in activities. With it, the selected activities are only executed after the page is fully loaded. Unfortunately, in this case the page navigating to the topic was not part of the page loading but happened after it. Because of that, the only solution I could come up with was adding an artificial two-second delay before the click activity for the edit dropdown, so that the page would have time to shift before the robot moved on.

After fixing that problem I ran the process again, this time with 100 courses, or 20 of each template. The execution took 1 hour and nine minutes, or around 41 seconds per course. Out of 100 courses, not even one had errors.

5.2 Case two: Data scraping from YTJ

For testing this process, I gathered a sample of 100 business IDs to scrape the data from. I wanted as much variance as possible, because the process for each business ID would not really change between repetitions, so repeating the process for the same business ID would be unlikely to yield different results.

This is because the process mainly happens on one page, the one where the data is scraped from, and that page will only change when searching for different business IDs.

As I began testing for this process, it quickly became apparent that something was wrong, as almost every business ID ended in an error. Something I had not noticed during the development, probably as a result of using too small of a sample as examples, was that most companies ended up not having either phone number or street address listed. Not having something listed meant that those rows were completely omitted from the table on the page, causing the robot to throw an error when scraping that row. However, this was easily fixed by configuring the robot not to throw an error from the activities that scraped those rows in the table, meaning that data from them was left empty if not found. That configuration in UiPath could be defined for each activity in its properties.

It is debatable whether this is a good solution, since now there will not be errors for those rows even if there is a legitimate problem with the scraping. That happening in this case is extremely unlikely though, as when the scraping for these rows happens, it is already established that the table exists by the previous activities using the same table.

With the robot configured not to throw errors for missing phone numbers or addresses, I ran the process for the 100 business IDs. The execution time was six minutes and 38 seconds, or approximately four seconds per business ID. As for errors, there were none, and the Excel had all the scraped data correctly saved. A sample of the Excel containing the data can be found in Figure 12.

	A	B	C	D	E	F
1	Trade name	Company form	Home munic	Main line of business	Address	Phone
2	Nokia Oyj	Public limited company	HELSINKI	Activities of head offices (70100)	Karaportti 3 02610 ESPOO	010 4488000
3	Neste Oyj	Public limited company	ESPOO	Manufacture of refined petroleum products (19200)	Keilaranta 21 02150 ESPOO	
4	Kesko Oyj	Public limited company	HELSINKI	Non-specialized wholesale (46901)	Sörnäistenkatu 2 00580 HELSINKI	
5	Stora Enso Oyj	Public limited company	HELSINKI	Manufacture of paper and paperboard (17120)		
6	UPM-Kymmene Oyj	Public limited company	HELSINKI	Manufacture of paper and paperboard (17120)	Alvar Aallon katu 1 00100 HELSINKI	
7	KONE Oyj	Public limited company	HELSINKI	Manufacture of lifting and handling equipment (28220)	Kartanontie 1 00330 HELSINKI	
8	Suomen Osuuskauppojen K	Cooperative	HELSINKI	Non-specialized wholesale (46901)	Fleminginkatu 34 00510 HELSINKI	10768011
9	Outokumpu Oyj	Public limited company	HELSINKI	Activities of holding companies (64200)	Salmisaarenranta 11 00180 HELSINKI	
10	Sampo Oyj	Public limited company	HELSINKI	Activities of holding companies (64200)		
11	UPM Sales Oy	Limited company	HELSINKI	Wholesale of other intermediate products (46760)	Åkerlundinkatu 11 B 33100 TAMPERE	
12	Metsäliitto Osuuskunta	Cooperative	HELSINKI	Logging (02200)	Revontulenpuisto 2 02100 ESPOO	
13	Wärtsilä Oyj Abp	Public limited company	HELSINKI	Manufacture of engines and turbines, except aircraft, vehicle	JOHN STENBERGIN RANTA 2 00530 HELSINKI	
14	Fortum Oyj	Public limited company	ESPOO	Activities of head offices (70100)	Keilalahdentie 2-4 02150 ESPOO	
15	North European Oil Trade O	Limited company	HELSINKI	Wholesale of liquid and gaseous fuels (46711)	Urho Kekkosen katu 5 C 00100 HELSINKI	
16	Outokumpu Stainless Oy	Limited company	TORNIO	Manufacture of basic iron and steel and of ferro-alloys (24100)	Terästie 95490 TORNIO	
17	Cargotec Oyj	Public limited company	HELSINKI	Manufacture of lifting and handling equipment (28220)	Porkkalankatu 5 00180 HELSINKI	
18	Valmet Oyj	Public limited company	HELSINKI	Activities of head offices (70100)	Keilasatama 5 02150 ESPOO	106720000
19	Konecranes Abp	Public limited company	HYVINKÄÄ	Manufacture of lifting and handling equipment (28220)	KONEENKATU 8 05830 HYVINKÄÄ	2042711
20	Huhtamäki Oyj	Public limited company	ESPOO	Activities of head offices (70100)	Revontulenkuja 1 02100 ESPOO	
21	Metso Oyj	Public limited company	HELSINKI	Activities of head offices (70100)	Toolönlahdenkatu 2 00100 HELSINKI	
22	Amer Sports Oyj	Public limited company	HELSINKI	Activities of holding companies (64200)	Konepajankuja 6 00510 HELSINKI	020 7122500
23	Finnair Oyj	Public limited company	HELSINKI	Scheduled air transport (51101)	Tietotie 9 01530 VANTAA	
24	Kemira Oyj	Public limited company	HELSINKI	Manufacture of other chemical products n.e.c. (20590)	Energiakatu 4 00180 HELSINKI	
25	Caverion Oyj	Public limited company	HELSINKI	Mechanical and process engineering design (71127)	Salmisaarenaukio 2 00180 HELSINKI	

Figure 12. A sample of the Excel with the scraped data

6 Conclusion

The primary goal of this thesis was to explore the web automation features of Robotic Process Automation through theory and practical use cases. From the perspective of that goal, I feel the thesis is successful. I am especially satisfied with the results of the implementation and testing of the use cases, as they ended up showcasing the different aspects of RPA quite well. They also had some parts that could be focused on for a more grass roots level analysis of web automation specific issues. The testing for them went smoothly, and to be honest, after the development I was not expecting too many problems with it either. Even though successful and mostly without problems, the tests did also have a few errors that could not be predicted at the implementation phase which is typical for RPA and therefore useful to be able to include in this thesis.

As for the secondary goal, I hope that this thesis is as easily digestible as I planned for it to be. I cannot speak for that myself, but in the theoretical part I think I was able to cover most of the essential aspects of RPA for the reader to be able to follow the practical part without prior knowledge of the subject. Also, the use cases being as straightforward as they were, I think anyone should be able to follow their process, implementation, and testing as well.

To be somewhat self-critical though, the use cases could have been more complex and had larger scopes. When planning them, I was not totally sure of what to prioritize, mainly whether to keep them simple and understandable or to expand them in order to showcase more features and pain points. I ended up leaning more towards simple and easy to understand, and as a consequence they did not demonstrate as many problems or interesting aspects as I had originally thought and hoped they would. This of course is not something one can completely plan for and could be flipped around to say that the product as well as the target applications worked better than expected.

Other than expanding the scope of the use cases, what I might have done differently in hindsight, is to give more consideration for free and open source solutions. With solutions like Robot Framework becoming available, their viability would perhaps be more interesting to study than the already established solutions. Also, licensing costs for the top products are not cheap and they are always a commitment of sorts to the product and RPA itself. Eliminating them with the use of free solutions might make companies that would not even consider RPA give it a chance.

To end with, RPA is obviously a promising business with a huge amount of money being funneled into it, so its development will be interesting to follow. Integrations into machine learning and AI solutions becoming more available in the future will hopefully allow RPA to automate not only rules-based processes but also more complex ones. The current solutions already seem extremely capable for automating simple processes like the ones presented in this thesis, and with the companies developing them raising hundreds of millions of capital, the evolution of the technology will likely be fast.

References

- Automation Anywhere. 2018. <https://www.automationanywhere.com/company/press-room/6145-automation-anywhere-raises-250-million-reaching-a-1-8-billion-valuation-in-one-of-the-largest-series-a-financing-rounds>. 1.11.2018
- AppliedAi. 2018a. <https://blog.appliedai.com/top-robotic-process-automation-rpa-benefits/>. 14.11.2018
- AppliedAI. 2018b. <https://blog.appliedai.com/rpa-implementation/>. 14.11.2018
- Arrow Digital. 2018. <https://www.arrowdigital.com/insights/2017/09/from-qa-to-rpa-an-unlikely-origin-story>. 30.10.2018
- Deloitte. 2017. Deloitte Global RPA Survey 2017
- Everest Group. 2018. Robotic Process Automation (RPA) Annual Report 2018 – Creating Business Value in a Digital-First World
- Forrester. 2018. The Forrester Wave™: Robotic Process Automation, Q2 2018
- Forrester. 2017. The Forrester Wave™: Robotic Process Automation, Q1 2017
- Microsoft. 2018. <https://docs.microsoft.com/en-us/dotnet/framework/windows-workflow-foundation/>. 3.11.2018
- Moodle. 2018. <https://moodle.net/stats/>. 13.10.2018
- Siili Solutions. 2018. <http://robotframework.org/rpa/>. 28.8.2018
- TechCrunch. 2018. <https://techcrunch.com/2018/09/18/ui-path-lands-225m-series-c-on-3-billion-valuation-as-robotics-process-automation-soars/>. 2.11.2018
- UiPath. 2018a. <https://www.uipath.com/press-room/uipath-raises-153-million-series-b>. 18.8.2018
- UiPath. 2018b. <https://www.uipath.com/press-room/uipath-leads-rpa-market-with-unmatched-customer-adoption>. 18.8.2018
- UiPath. 2018c. <https://www.uipath.com/community>. 18.8.2018
- UiPath. 2018d. <https://www.uipath.com/blog/the-technical-journey-of-uipath>. 19.8.2018
- UiPath. 2018e. <https://studio.uipath.com/docs/about-selectors>. 19.8.2018
- UiPath. 2018f. <https://www.uipath.com/studio>. 19.8.2018
- UiPath. 2018g. <https://www.uipath.com/robot>. 19.8.2018
- UiPath. 2018h. <https://www.uipath.com/orchestrator>. 19.8.2018
- UiPath. 2018i. <https://www.uipath.com/blog/rpa-ai-success-stories>. 9.9.2018