Towards Automated Testing of RPA Implementations

Marina Cernat*
marina.cernat@drd.unibuc.ro
Dept. of Computer Science
University of Bucharest
Romania

Adelina Nicoleta Staicu*
adelina.staicu@cegeka.com
Cegeka
Romania

Alin Stefanescu*
alin@fmi.unibuc.ro
Dept. of Computer Science
University of Bucharest
Romania

ABSTRACT

Robotic Process Automation (RPA) is a technology that has grown tremendously in the last years, due to its usability in the area of process automation. An essential part of any software development process is quality assurance, so testing will be very important for RPA processes. However, the classical software techniques are not always suitable for the RPA software robots due to the mix of the graphical description of the robots and their implementations. In this short paper, we describe the state of the practice for testing of software robots and propose some ideas of test automation using model-based testing.

CCS CONCEPTS

• Software and its engineering → Software testing and debugging; **• Applied computing → Enterprise applications**.

KEYWORDS

Test automation, Robotic Process Automation (RPA), Model-based testing, RPA testing

ACM Reference Format:
Marina Cernat, Adelina Nicoleta Staicu, and Alin Stefanescu. 2020. Towards Automated Testing of RPA Implementations. In *Proceedings of the 11th ACM SIGSOFT International Workshop on Automating TEST Case Design, Selection, and Evaluation (A-TEST '20), November 8–9, 2020, Virtual, USA.* ACM, New York, NY, USA, 4 pages. https://doi.org/10.1145/3412452.3423573

1 INTRODUCTION

Robotic Process Automation (RPA) [28] is nowadays the fastest-growing segment in the enterprise marketplace, with up to 40% annual growth rates and an increase of market valuation from $1.4 billion in 2019 to an estimated $27 billion in 2027 [15]. This tremendous success is due to the promise of affordable automation of enterprise processes to save time and money by reducing repetitive tasks. RPA is the technology that allows the configuration of a so-called "software robot", to emulate the actions that are generally made by a human who interacts with a digital system while executing a business process. The software robot interacts with the system at the UI level, mimicking the actions of a user by clicking through the applications' interfaces or filling forms with data copied from places that are accessible at the front-end level.

Along with great opportunities, RPA involves also many challenges [23] and risks, with estimates that up to 50% of RPA projects fail [16]. In particular, since the RPA is a new paradigm, its development lifecycle is still maturing. One important aspect is quality assurance, which is a time-consuming task upon which the success of a project depends.

Testing is the most used technique to verify that the implemented robots are working according to the specifications and without too many errors. However, currently, most of the RPA testing is done manually, with very few tools to support this process.

In this short paper, we want to open a discussion on the topic of RPA testing and, more precisely, how we could improve its state-of-the-practice with the state-of-the-art research in testing. Conversely, the study of automation in RPA testing can provide new interesting research topics - cf. [3].

2 ROBOTIC PROCESS AUTOMATION

2.1 RPA in a Nutshell

Robotic Process Automation is the technology that provides tools to automate actions and processes executable at the UI level. The first-class citizen in RPA is the software robot (sometimes called also a "bot"), which can perform all the basic actions that a human also performs through the user interface of the available applications, including legacy software. More precisely, the robot can click buttons, copy data from and to various applications, but also use APIs, if necessary. The processes amenable to RPA implementations are those that are repetitive enough to justify an investment in automation, but not structured and complex enough to require a classical software development project [28].

RPA providers offer platforms to define and operate such software robots. The mature RPA frameworks contain at least the following three components: a component to describe or model the robots (this is usually done visually), a component to execute and integrate the robots in the environment and existing applications, and, finally, a component, usually called an orchestrator, that deploys the robots, schedules and monitors their execution. On top of that, the advanced RPA solutions also offer artificial intelligence (AI) capabilities, such as natural language processing (NLP), machine learning, and computer vision, to better process the textual or visual inputs of the robot. Also, the robots may be attended (requiring human input from time to time) or unattended (running independently). They can run locally, in the cloud, or in a virtual environment.

[*] All the authors had an equal contribution to this paper.

A-TEST '20, November 8–9, 2020, Virtual, USA

ACM ISBN 978-1-4503-8101-7/20/11...$15.00
https://doi.org/10.1145/3412452.3423573

The RPA market is very dynamic and rather fragmented. However, over the last years, the following three RPA tool providers - UiPath (https://www.uipath.com), Automation Anywhere (https://www.automationanywhere.com), and Blue Prism (https://www.blueprism.com) - were designated as leaders in the field by independent market research companies such as Gartner [13] and Forrester [10]. Based on the recent round of investments, UiPath has reached the status of "decacorn", i.e., being valued at more than $10 billion. Besides those companies, several contenders are offering RPA solutions: WorkFusion (https://www.workfusion.com), EdgeVerve (https://www.edgeverve.com), Kofax (https://www.kofax.com), Softomotive (https://www.softomotive.com) (acquired by Microsoft), Kryon (https://www.kryonsystems.com), or new entrants such as Leapwork (https://www.leapwork.com) or Tricentis (https://www.tricentis.com).

Even though the above RPA offerings are similar in scope and functionalities, their implementations and design choices are rather diverse. Therefore, in the rest of the paper, we will focus on the UiPath platform, which is arguably the main RPA tool provider, based on company size and valuation, market share as well as breadth and depth of the solution. Moreover, from our research, it has the most advanced RPA testing support, thus making it a good candidate for our investigation of RPA testing and its automation. Furthermore, we are more knowledgeable in UiPath technologies (the 2nd author is an UiPath RPA developer with 2 years of industrial experience). Yet, we will also comment on other tool providers in certain contexts.

2.2 RPA in UiPath

In this subsection, we will give more technical details about the RPA framework of UiPath.

The component where the robots are designed is called UiPath Studio [21]. It is based on Microsoft Workflow Designer [7], which is an IDE for the graphical construction of Windows Workflow Foundation [11] applications. The order in which the activities may be performed by a robot can be modeled in UiPath Studio with three main types of workflow diagrams [6]: sequence, flowchart, and state machine (plus an extra one for exception handling). The sequence is a simple linear representation of the activities that follow one after another (allowing also an if activity). The flowchart adds more flexibility through decisions and arrows between any activities. It is similar to UML activity diagrams, but also BPMN models. Finally, the state machine is even more expressive than a flowchart, by allowing conditional transitions. They are similar to the classical UML state machines. The above constructs can also contain data in the form of diagram arguments and local variables. Also, very importantly, all the diagrams can be hierarchically embedded into one another, e.g., we may have a sequence of flowcharts that contain local state machines in certain nodes.

The workflow is created by drag-and-drop from the existing basic activities (e.g., code snippets written in C#) or other existing workflows. The basic activities can, for instance, read and write data from and into several formats (PDF, Excel, Word, common databases, desktop or web-based applications), can work with all types of variables (the UiPath Studio is built upon the .NET framework), can create reports, handle mouse clicks and keyboards strokes and even convert handwriting, printed text, or text inside images into machine-readable text using Optical Character Recognition (OCR). These activities can be either directly programmed or, more conveniently, generated using a UI recording feature of the tool.

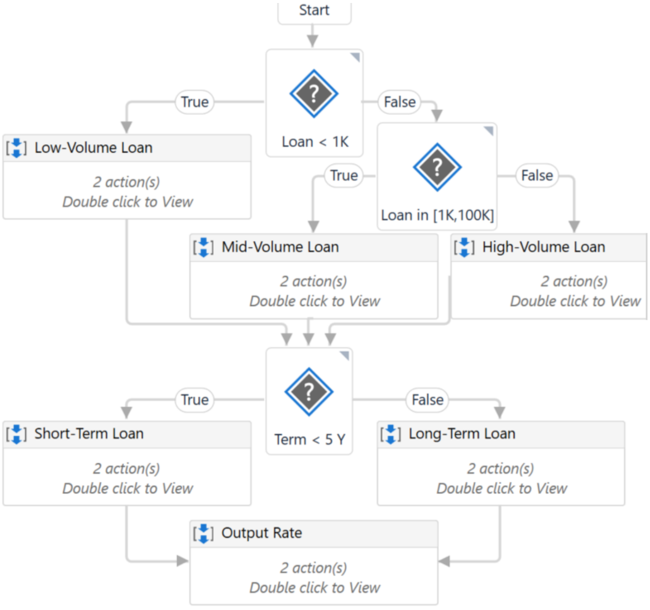

Figure 1: A simple robot

Also, RPA can handle complex tasks using advanced plugins, including machine learning, computer vision, or cognitive automation, working even with virtual or remote environments. Regarding virtual desktop interfaces (VDIs), UiPath was among the first to provide UI automation for technologies like Citrix, managing to solve the problem of obtaining the underlying properties of UI elements (buttons, text fields, etc.), the difficulty here being that VDIs provide only an image of the remote desktop.

The other RPA tool providers offer similar features and models, each of them having a focus on one aspect or another. Due to space limitations, we will not cover them here.

3 EXAMPLES OF RPA ROBOTS

3.1 A Simple Loan Example

For a better understanding of how RPA works, let us consider a simple process (taken over from the UiPath documentation [22]) that calculates the type of a bank loan based on the value of the amount and the duration of the loan.

In the flowchart in Fig. 1, named Create_Loan_Process, we have three decisions (depicted with a '?' and a condition). The first two ones reference the amount of the loan and change the direction of the flow on one of the branches corresponding to its category: low, medium, or high. The last decision refers to the duration of the loan. If the duration is less than, for example, 5 years, then the loan is short-term, otherwise, it is long-term.

3.2 A More Complex Example

In the retail industry, rule-based, high volume processes are often automated. One common B2C (Business-to-Consumer) process in retail is the refunding of the return orders, which can involve a lot of human resources, especially in peak periods, when sales increase. Robots can take over the refunding tasks, helping companies to save time, to increase compliance, and to free-up resources. The input of the robot is represented by a report in web format containing information like the number of a sales order, the data of the customer accounts, the returning date, etc. The robot logs into a CRM system and searches for every sales order number copied previously from the input report. Then, for each refund request, it establishes what items from the order need to be refunded and calculates the amount of money to refund. If needed, the robot applies a discount in case the customer had a voucher. Once the sum is calculated, the robot continues the process by accessing a second application - a payment platform, where it will send the total refund amount to the entitled customer. The status of every action is written in an output Excel file, including the sales order numbers and the refunded amounts.

4 CURRENT RPA TESTING

In this section, we discuss the state-of-the-practice in RPA testing. To prevent confusion, by RPA testing [25], we mean the testing of an RPA implementation or a robot. We mention this because sometimes RPA may be used as a tool for (UI) application testing [4, 24]. In fact, there is a common history of RPA and (UI) testing [9] in the sense that some of the current RPA tools evolved from automated UI testing tools as it was the case for companies such as Automation Anywhere, Tricentis, and Leapwork.

Even though the RPA development is a nascent domain, most providers have lifecycle management [18] in place for enterprise-ready implementations, with testing playing an important role. However, testing an RPA implementation can be a tedious task. Once the development phase is ready, testers or developers need to follow the defined requirements and manually create test cases to test the functionality of the robots. Both unit and integration testing are performed. The test suite contains possible scenarios that could happen while running a robot, following all the key steps and rules defined in the analysis phase. The test cases are mostly written in an Excel file, containing pre-conditions, post-conditions, the expected result, and the actual result, all the fields being manually provided by business analysts. It is worth noting that the lifecycle contains also complementary steps such as static model validation through a Workflow Analyzer [1] and also debugging and code review support.

To manage the testing lifecycle, UiPath launched this year a solution named UiPath Test Suite [22], which is among the first of its kind in the RPA domain. This solution offers test management by organizing the test suites, test execution, and test reporting. It supports RPA testing, UI application or mobile testing, and API testing through data-driven test case design.

For RPA testing, UiPath has created a test case template as a dedicated sequence workflow that contains another three subsequences named *Given*, *When*, *Then*, corresponding to test preparation, test execution, and test evaluation/assert, respectively. When testing a robot, a test suite of multiple test cases will be created, covering various scenarios. A feature of the UiPath Test Suite is that after executing the test suite, the tester gets the test coverage as a percent of covered activities of the robot, highlighting them visually.

5 TOWARDS AUTOMATED RPA TESTING

While the test execution and test management are automated, as seen in the previous section, the test design is a manual task and we are not aware of any tool automating it through test case or test data generation. Since the robots are implemented based on visual models such as flowcharts or state machines, the idea of applying model-based testing techniques [26] comes in naturally.

Model-based testing (MBT) is an approach that uses a model of the system under test to automatically generate test cases. While the theoretical foundations were laid more than 40 years ago, the field started to gain traction 20 years ago, with several tools developed and many papers published [27]. Although the testing industry showed interest and started to experiment with MBT, the adoption is still rather slow. While the theory and even tooling are mature, there are several hurdles in the industrial MBT application: lack of models, high computational complexity and state-space explosion, difficulty of integration in heterogeneous landscapes, non-trivial definition of test selection criteria, etc.

If we look at the robot from Sect. 3.1, it is clear that an MBT textbook algorithm can easily generate a test suite that covers all activities and also all paths, with 6 pairs of values for (*loan*, *term*), with *loan* taking 3 values and *term* 2 values (e.g., *loan* in {500, 1500, 100500} and *term* in {2, 12}.

Since it is not practical to reimplement test generation algorithms, we plan to use an off-the-shelf tool to generate test cases from RPA models. For that, we have to transform the models into the input format accepted by an MBT tool, then use its test generation engine and transform the generated test cases back into the test case format accepted by the RPA test management framework. However, the portability of models is one of the well-known challenges of MBT adoption as mentioned in [27]. In our case, as specified in Sect. 2.2, the UiPath graphical models are based on Windows Workflow Foundation (WF).

We first searched for tools related to testing for WF models and we found only: [17], a tool that provides test execution and coverage visualization capabilities, similar to the recent UiPath Test Suite solution, and [14], which builds upon [17] adding a unit testing dimension. Slightly related work is also [5], which considers a test-driven development process of robots. However, none of them deals with test generation.

Next, we searched for tools exporting WF models into classical UML or BPMN diagrams or any other visual diagram type used as input by MBT tools, but we have found none. Therefore, we decided to implement the model transformation ourselves. We wrote a C# program that recursively extracts the elements of the model from its XAML format (Extensible Application Markup Language) and we are in the process of mapping those elements to both UML and BPMN diagrams and we are investigating best ways to translate specific aspects regarding to data (types, scopes) or the hierarchical embedding of models (a sequence may contain inside a node an embedded flowchart, which may contain inside a statechart, which may have embedded a sequence, and so on). Of course, we can deal with a simple example such as the one in Sect. 3.1, which has a flat structure, 2 simple variables, and 3 decisions, but the industrial RPA implementations are much more complex.

At the same time, we are reviewing all the current MBT solutions, to check their input formats and other peculiarities. On the one hand, there are open source tools such as GraphWalker (http://graphwalker.github.io), a rather simple tool to generate paths in a graph, which must be complemented by a constraint solver for test data generation. On the other hand, there are very powerful commercial MBT offerings from companies such as Conformiq (https://www.conformiq.com) and Smartesting (https://www.smartesting.com), which can deal with a variety of input formats, UML and BPMN, or more lightweight tools based on BPMN such as Yest [8] or ETAP-Pro [20].

6 FINAL DISCUSSION AND FUTURE WORK

The main goal of this short paper is to bring RPA testing to the attention of the academic community. There is a good potential to transfer and adapt research results to improve the automation of RPA testing - cf. [2]. We think that this is a worthwhile endeavor given the practical impact it can have in a fast-growing enterprise domain. Even though the RPA concepts can be mapped to existing testing research methods, there are still many non-trivial challenges to be addressed [16, 19, 23].

We are at the beginning of our applied research project for RPA testing, but we identified several paths that we plan to explore. First, we will focus on finishing the UiPath model transformation and experiment with an open-source MBT tool, but also with commercial offerings from Conformiq and Smartesting (via academic licenses). Then, we will transform the generated tests back into the UiPath Test Suite to be executed. We will do this for a couple of medium complexity industrial models to which we already have access. If successful, we will distribute our work as an open-source MBT plugin in the UiPath app marketplace and gather further feedback from UiPath developers and testers.

Using the lessons learned from this experiment, we will try to replicate the approach with similar models from other RPA providers (Automation Anywhere, Blue Prism, etc). For that, we will only have to adapt the model transformation to the new model format (these could be offered as MSc student projects). We already identified a couple of models exportable to XML.

Finally, in a bolder attempt, we will investigate the test automation problem in the context of hyperautomation, a new buzzword, listed by Gartner [12] as no. 1 in a top 10 list of technological trends. Hyperautomation means RPA enhanced by AI, NLP, analytics, process mining, and other technologies that enable scenarios where human decision-making is required, involving volatile environments and cognitive capabilities. This will require a clever combination of MBT with AI testing, which is also an emerging testing domain.

ACKNOWLEDGMENTS

This work was supported by a grant of the Romanian Ministry of Research and Innovation CCCDI-UEFISCDI, project no. 17PCCDI/2018. We also thank Ingo Philipp, Vice President at UiPath, for inspiring discussions on the topic of the paper.

REFERENCES

[1] UiPath Workflow Analyzer. 2020. https://docs.uipath.com/studio/docs/about-workflow-analyzer.

[2] A. Arcuri. 2018. An experience report on applying software testing academic results in industry: we need usable automated test generation. *Empirical Software Engineering* 23, 4 (2018), 1959–1981.

[3] L. Briand, D. Bianculli, S. Nejati, F. Pastore, and M. Sabetzadeh. 2017. The Case for Context-Driven Software Engineering Research: Generalizability Is Overrated. *IEEE Software* 34, 5 (2017), 72–75.

[4] M. Cernat, A.-N. Staicu, and Stefanescu A. 2020. Improving UI Test Automation using Robotic Process Automation. In *15th Int. Conf on Software Technologies (ICSOFT'20)*. SciTePress, 260–267.

[5] C. Cewe, D. Koch, and R. Mertens. 2018. Minimal Effort Requirements Engineering for Robotic Process Automation with Test Driven Development and Screen Recording. In *Proc. of BPM'18 Workshops (LNBIP)*, Vol. 308. Springer, 642–648.

[6] UiPath Workflow Design. 2020. https://docs.uipath.com/studio/docs/workflow-design.

[7] Microsoft Workflow Designer. 2016. https://docs.microsoft.com/en-us/visualstudio/workflow-designer.

[8] B. Elodie, F. Ambert, B. Legeard, and A. Bouzy. 2018. Lightweight Model-Based Testing for Enterprise IT. In *Proc. of ICST'18 Workshops*. IEEE, 224–230.

[9] Forrester Research. 2020. RPA And Test Automation Are More Friends Than Foes - But You Can't Extend Test Automation Technology To RPA And Vice Versa. Research report, no. RES159815.

[10] Forrester Research. Q4 2019. Forrester Wave™: Robotic Process Automation. Market research report, no. RES147757.

[11] Microsoft Windows Workflow Foundation. 2017. https://docs.microsoft.com/en-us/dotnet/framework/windows-workflow-foundation.

[12] Gartner Research. 2019. Top 10 Strategic Technology Trends for 2020. Research report, no. G00432920.

[13] Gartner Research. 2020. Magic Quadrant for Robotic Process Automation Software. Market research report, no. G00441474.

[14] U. Goljat and M. Heričko. 2009. Testing workflow-enabled applications. In *Proc. of 20th Central European Conf. on Inform. and Intelligent Systems*. 349–354.

[15] Grand View Research. 2020. Robotic Process Automation Market Size and Share Report, 2020-2027. Market research report, no. GVR-1-68038-145-0.

[16] L.-V. Herm, K. Fuchs, A. Helm, A. Hofmann, F. Imgrund, Janiesch C., and A. Winkelmann. 2020. A Consolidated Framework for Implementing Robotic Process Automation Projects. In *18th Int. Conf. on Business Process Management (BPM'20) (LNCS)*, Vol. 12168. Springer, 471–488.

[17] A. Lechner and R. Breu. 2009. Workflow Inspector - A Test Tool for Microsoft Workflow Foundation. In *Proc. of ICST'08*. IEEE, 498–501.

[18] UiPath Automation Lifecycle. 2020. https://docs.uipath.com/studio/docs/automation-lifecycle.

[19] P. Noppen, I. Beerepoot, I. van de Weerd, M. Jonker, and H. Reijers. 2020. How to Keep RPA Maintainable?. In *18th Int. Conf. on Business Process Management (BPM'20) (LNCS)*, Vol. 12168. Springer, 453–470.

[20] A. Paiva, N. Flores, J. Faria, and J. Marques. 2018. End-to-end Automatic Business Process Validation. In *ANT/SEIT 2018 affiliated workshops (Procedia Computer Science)*, Vol. 130. Elsevier, 999–1004.

[21] UiPath Studio. 2020. https://www.uipath.com/product/studio.

[22] UiPath Test Suite. 2020. https://www.uipath.com/product/test-suite.

[23] Syed R. *et al.* 2020. Robotic Process Automation: Contemporary themes and challenges. *Computers in Industry* 115 (2020), 15 pp.

[24] UiPath Application Testing. 2020. https://docs.uipath.com/studio/docs/application-testing.

[25] UiPath RPA Testing. 2020. https://docs.uipath.com/studio/docs/rpa-testing.

[26] M. Utting and B. Legeard. 2007. *Practical Model-Based Testing - A Tools Approach.* Morgan Kaufmann.

[27] M. Utting, B. Legeard, F. Bouquet, E. Fourneret, F. Peureux, and A. Vernotte. 2016. Recent Advances in Model-Based Testing. *Advances in Computers* 101 (2016), 53–120.

[28] W. van der Aalst, M. Bichler, and A. Heinzl. 2018. Robotic Process Automation. *Business & Information Syst. Eng.* 60, 4 (2018), 269–272.