Towards a Method for Automated Testing in Robotic Process Automation Projects

J. Chacón-Montero, A. Jiménez-Ramírez, J.G. Enríquez
Departmento de Lenguajes y Sistemas Informáticos
Universidad de Sevilla
Seville, Spain

jesus.chacon@iwt2.org, ajramirez@us.es, jgenriquez@us.es

***Abstract*—The Robotic Process Automation (RPA) paradigm has received increasing attention in recent years. It enables task automation by software components which interact with user interfaces in a similar way to that of humans. An RPA project follows a similar lifecycle as a software project. However, in certain contexts (e.g., business process outsourcing, BPO), a testing environment is not always available. Thus, deploying the robots in the production environment entails high risk. To mitigate this risk, an innovative approach to automatically generate a testing environment and a test case for an RPA project are described. The activities of the humans whose processes are to be robotized are monitorized and an UI log (i.e., a sequence of screen captures, mouse and key actions) is confirmed. On the one hand, the test environment is generated as a fake application, which mimics the real enviroment by leveraging the UI log information. To this end, the control flow of the application is governed by an invisible control layer which decides which image to show depending on the interface actions that it receives. On the other hand, the test case checks whether the robot can reproduce the behaviour of the UI log. A prototype has been constructed and tested in a controlled scenario. Promising results have been obtained and a number of limitations to be addressed have been identified such that it may be applied in more realistic domains.**

***Index Terms*—automated testing, robotic process automation**

I. Introduction

Robotic Process Automation (RPA) is a software solution for the creation of programs that mimic the behaviour of human workers when performing repetitive and structured tasks with information systems (ISs) [6], [21], [24]. This solution has been applied in many industries and contexts. However, several authors have acknowledged that the best candidates to conduct a successful RPA project are the processes within the back offices of a company [7], [20] since they (1) are highly frequent, (2) lack excessive exception control, (3) require limited cognitive effort, and (4) are prone to human errors [6].

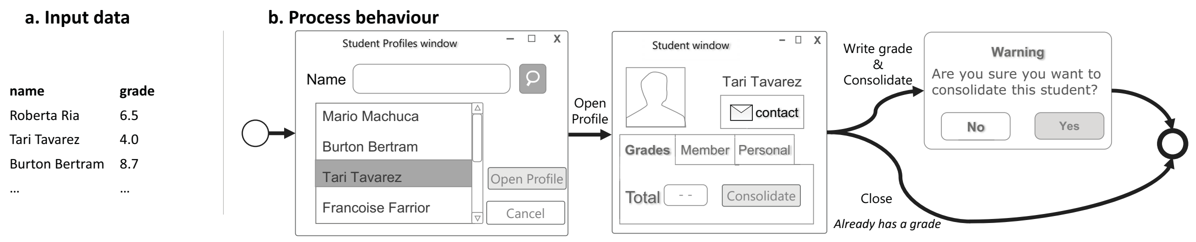
There are many vendors who offer out-of-the-box solutions to deal with RPA, such as BluePrism [5], UIPath [22], and WorkFusion [25]. In general, such tools share a common RPA lifecycle although each tool provides different support to each phase. The lifecycle starts with the analysis of the candidate process for automation. The processes are then designed to contain the actions, data-flow, etc. that must be coded. Subsequently, robots are constructed in accordance with to the design. They are then deployed in individual environments (e.g., virtual machines) to perform their tasks. During their deployment, the robots are controlled and monitored in their operation (e.g., to start new robots, stop them in case of serious errors, etc). Finally, the performance and error cases of the robots are evaluated to enable a new analysis for the enhancement of the robots. It is important to bear in mind that the testing phase is missing since it heavily depends on each RPA project.

In the traditional software lifecycle, testing is performed in a testing environment before deployment in the production environment. Unlike traditional software, such a testing environment is seldom available in RPA, which entails a high risk for deployment. In these situations, companies about to adopt RPA start performing the processes with human workers rather than robots. For this reason, such employees receive training to learn how to manage cases, and they also learn through solving the exceptions that occur on a daily basis. In a parallel way, the RPA project starts by automating the simplest cases by leveraging the knowledge that the human workers have generated. In this paper, we propose leveraging such information to automatically test the constructed robots.

To this end, a method to automatically generate a testing environment and a test case for RPA projects is proposed. The first step that must be executed involves controlling the input data that the user needs to work (e.g., a list of emails with a specific subject, and a spreadsheet file with client information). The computers of the human workers who are performing the processes to robotize and gather this behaviour in the form of a UI log [13] are monitored. This log contains the user interactions with screen captures for each case of the input data. The test environment can then be generated as a fake application, which mimics the real IS, that consists of a set of images and an invisible control layer over such images. This invisible control layer is in charge of capturing robot inputs (i.e., mouse clicks and keystrokes) and deciding the next image to show depending on said inputs. In addition, this layer controls the state of the test, i.e., *pass* or *fail*.

Once the generated testing environment is ready, the robots that are constructed are deployed therein and stimulated with the input data of the test case (this step lies outside the scope of this paper). Finally, a test report is generated by the control layer.

A preliminary prototype has been evaluated in controlled



name	grade
Roberta Ria	6.5
Tari Tavarez	4.0
Burton Bertram	8.7
...	...

Fig. 1. Running example

scenarios and the results seem promising. This innovative idea opens several research lines regarding the current limitations.

The rest of the paper is organized as follows: Section II describes a running example that will be used to guide and illustrate the various points of this proposal. Section III contextualizes the proposal, and provides the details of the RPA paradigm and focuses on how it affects the aforementioned BPO context. Section IV presents the studies most closely related to the proposal of this paper found in the literature. Section V details the approach by taking the running example described in Section II as reference. Finally, Section VI summarizes the conclusions obtained after developing this approach and proposes several new research lines.

II. Running example

To contextualize the approach, this section describes a simplification of the process a teacher has to perform on her institution website for the consolidation of the results of the exams that she has marked. To this end, a set of student marks are needed (such as the spreadsheet in Figure 1a), which can be considered as the input data. Each student is dealt with as a different case. For each estudent, the teacher behaves as follows (Figure 1 b.). First, she looks for the student in the Student Profiles window. She then has to open the student profile and check whether this student already has a mark (e.g., other teacher did it before). If it is not the case, the mark has to be entered and then consolidated. A warning window is shown to confirm the action. In the case that the student already has a mark, the teacher simply closes the window. In both cases, the application returns to the student profile window.

id	timestamp	action	window
12	2018/01/10-0:4:32	Text "Tari Tavarez"	Student profile
13	2018/01/10-0:4:35	Left click on (120,205)	Student profile
14	2018/01/10-0:4:50	Left click on (10, 240)	Student
15	2018/01/10-0:5:05	Text "Burton Bertram"	Student profile
16	2018/01/10-0:5:09	Left click on (115,206)	Student profile
...	...	...	...

Fig. 2. An example interaction with the system

Figure 2 shows an excerpt of the interaction of the teacher with the system. As can be observed, in the interactions 12 - 14, the teacher is dealing with the case of the student "Tari Tavarez". She first enters the student name, then left-clicks on the coordinates 120, 205 (i.e., the "Open Profile" button), and finally closes the Student window (i.e., clicks on 10, 240) after having seen that this student already has a mark. The teacher then proceeds with the subsequent student case.

This simplified process is repetitive and is therefore a perfect candidate for robotization since the interactions with the user interface can be performed by a software robot. Following the construction of such a robot (which lies outside the scope of this paper), its behaviour has to be tested. However, the only platform available for said testing is the real-life system, and, testing in this environment therefore entails a high risk. For this reason, our approach shows how to generate a testing platform for the robotic projects.

III. Background

A. RPA

The ever-expanding RPA concept is built upon three main ideas: Robot, process, and automation [2]. While *robot* means whatever software machine can be programmed to execute a specific job, *process* is about a succession of steps to reach a particular objective or result. Lastly, *automation* means a transformation of a manual operation with the purpose of enabling it to be performed in an autonomous way.

Broadly speaking, a robot neither needs to be autonomous nor to follow a process (e.g., it could be commanded by a human). When those three elements join together, then RPA arises. Therefore, in this context, a robot is a software product created to perform a specific task. This task is a process that can be performed autonomously by the robot. The main objective is to minimize costs, provide agility, and improve the quality [3] of processes that have hitherto been performed by a human team [17].

For the construction of such robots, an RPA project must be conducted following a lifecycle which involves six main steps (cf. Figure 3): (i) analysis, (ii) design, (iii) development, (iv) deployment, (v) testing, and (vi) maintenance and operation. In the analysis step, various techniques are applied to successfully understand the business process to be robotized. In the design and development steps, the developers must define the process in order to then translate it and build the robot. This robot would carry out the process that has been formalized in the analysis step. Although a test environment is desirable for testing purposes and may be available in certain contexts, it should be borne in mind that general RPA contexts (e.g., the BPO context) are characterized as lacking a test environment and only the production environment is available for such purposes [9]. Therefore, when the robot is implemented, it is deployed in production and then tested to determine whether it is behaving correctly. Once checked, the robot is the maintained and operated. In this phase, its performance is measured and the robot can be stored, started, and/or replicated depending on its state and/or on the current demand. Once checked, the robot is maintained and it operates to measure its performance and to start, stop or replicate it in base of its state or the demand.

As the reader may have noticed, this lifecycle presents one main drawback: the testing phase is executed directly in the production environment which places production at high risk.

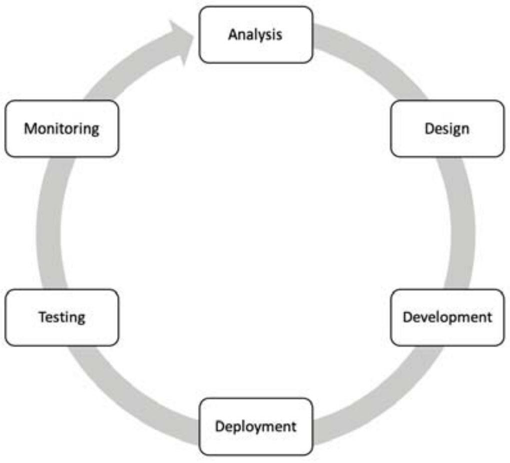


Fig. 3. Typical RPA lifecycle

B. RPA in the BPO Context

Several authors, through practical experience, acknowledge that the best process candidates to guarantee a successful RPA project are processes within the back office of a company [7], [20] since they are highly repeatable, with a clear workflow, require low cognitive effort, and prone to errors [6](e.g., the running example stated in Section II).

Robotic Process Automation becomes particularly challenging in BPO scenarios since processes are outsourced, in that, the back office and the ISs to interact with may be geographically dispersed. An outsourced process implies that the company remains unaware of how the process will be executed. The BPO company is responsible for deciding whether a robot should be developed for any particular process. However, when a robot is to be implemented, there are some cases where the robot development team have no access to a test environment.

In this context, a partnership with the Servinform S.A.[1] company has been established to improve the current RPA lifecycle. In Servinform, and any BPO setting in general, the back office consists of both human and robot teams. They

[1]Servinform is a Spanish BPO company with an IT consulting area.

TABLE I
EVENT ATTRIBUTES CAPTURED IN THE UI LOG

Name	Description
app_name	This attribute refers to the name of the application that is being used.
$focus$	This attribute indicates whether the application has just gained the focus.
$event\_type$	This attribute refers to the kind of events that can be produced $\{click, keystroke\}$.
$keystrokes$	This attribute refers to the sequence of keystrokes pressed when $event\_type$ is a keystroke.
$click\_type$	This attribute refers to the kind of click that has been produced $\{left, right, middle\}$ when $event\_type$ is $click$.
$click\_coords$	This attribute refers to the coordinates of the click when $event\_type$ is $click$.
$start\_ts$	This attribute refers to the start timestamp.
end_ts	This attribute refers to the end timestamp.
img_name	This attribute refers to the name of the file that contains the screen capture.

typically perform the following process when an outsourced process is to be managed (cf. Figure 4 (a)). First, the human workers are trained (cf. Figure 4 (b)) to perform such a process. These workers then start to deal with the real ISs according to their training (cf. Figure 4 (c)). It is at this moment when the RPA lifecycle starts. In order to leverage the knowledge that the human workers have acquired, their computers are monitored to obtain a log of the interactions with the user interfaces (i.e., a UI log) [13] (cf. Figure 4 (d)). Such a UI log consists of a stream of events, which contains clicks, keystrokes, screen captures and further relevant information (see Table I) for the UI log information proposed by Servinform. This log provides invaluable information for the analysis of the underlying outsourced process and the enhancement of a successful RPA lifecycle (cf. Figure 4 (f)). As a result, a robotic team is deployed to interact with the same ISs in parallel with the human team.

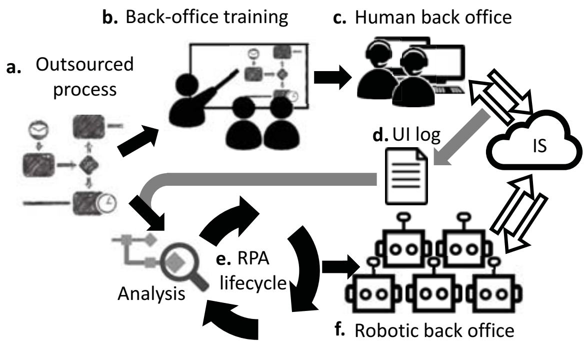


Fig. 4. RPA in the BPO

IV. Related Work

The scope of application of RPA is diverse, since it can be applied in environments such as: Front and Back-Office [8], [10], [19], the aforementioned BPO [1], [4], [10], health systems [16], [18], eCommerce or facial recognition [16] and finance [3], [11]. However, to the best of our knowledge, there are just a few proposals that speak about how to test the robots implemented.

In the work presented by Kämäräinen [9], the author explains the complicated structure that currently exists for the development of RPA. The author points out that, in the early stages of development, the robots work well on the development servers; however, the transition to the production phase becomes overly complicated. This is usually due to the fact that developers lack permission to access the test servers, which means they cannot test their implementations in the pre-production environment and therefore the clients are often requested to test the implemented robots.

Le Clair [12] summarizes a number of best practices and provides guidance for the prioritization of processes for the analysis phase. However, this work lacks any technical support for said phase. From a more technical point of view, Leopold et al. [14] propose the use of natural language processing and machine learning for the detection of candidate activities from the textual description of processes. In turn, Linn et al. [15] introduce the concept of desktop-activity mining. In this work, the authors combine monitoring techniques with process mining. However, it would not be feasible to apply this in complex settings, such as the BPO domain, since it requires access to actual UI elements. Similarly, Leno [13] suggests that using process mining for the discovery of local process models (i.e., frequent patterns) from UI logs may be useful in training robots. Furthermore, Van der Aalst et al. [23] describe how RPA may leverage techniques from the process-mining paradigm. However, none of these techniques directly deals with the testing phase.

V. Approach

This section describes our proposal to automatically generate a testing platform for an RPA project in the early phases of the RPA lifecycle.

As can be seen in Figure 5, this method changes the original lifecycle in certain new aspects. Essentially, it considers the inclusion of two new steps in the lifecycle: the test environment construction (cf. Section V-A) and the automatic testing (cf. Section V-B).

A. Test environment construction

The first step of our proposal consists of building a platform that mimics the behaviour of the real system with which the robots will interact. To this end, the UI log (cf. Table I), which is provided in the analysis phase, is used as the basis. As stated in Section III-B, this log corresponds to the interaction of a human performing the process that is to be robotized.

Example 1: Considering the running example and the interaction of Figure 2, Figure 6 shows the information of the corresponding UI log that would be needed to build the test environment.

Using the details of the actions and the screen captures of the UI log (cf. Example 1), the test environment is constructed as a UI controller with two main functionalities:

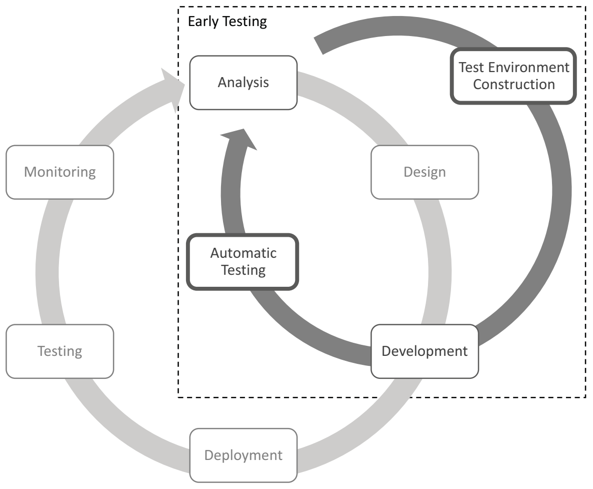
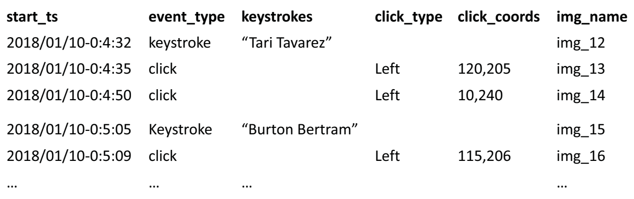


Fig. 5. Modified RPA lifecycle



start_ts	event_type	keystrokes	click_type	click_coords	img_name
2018/01/10-0:4:32	keystroke	"Tari Tavarez"			img_12
2018/01/10-0:4:35	click		Left	120,205	img_13
2018/01/10-0:4:50	click		Left	10,240	img_14
2018/01/10-0:5:05	Keystroke	"Burton Bertram"			img_15
2018/01/10-0:5:09	click		Left	115,206	img_16
...	...	...			...

Fig. 6. Running UI log

1) Showing views: The environment will show a succession of views (i.e., images) working as the background of the application. Such views are shown in the same order as they appear in the UI log.
2) Capturing events: In front of the views, an invisible layer captures all the UI events (i.e., clicks and keystrokes). This layer also acts as a controller in that it checks whether the captured events are similar and in the same order as the events which appear in the UI log.

With the background and the invisible layer, the original application can simulated.. Therefore, this platform can be used to perform tests in an automatic way.

B. Automatic Testing

When the developer team completes the construction of robots, our proposal can be used to automatically test these robots before continuing with the lifecycle. To this end, a robot must be deployed to interact with the test environment described above with the same input data as that used while creating the UI log (cf. Figure 7).

The automatic testing process, which is described in Alg. 1, receives the aforementioned UI log and the UI controller (i.e,. the test environment). The objective of this algorithm is to compare the actions stored in the UI log against each action that a robot performs. The result of such a comparison produces a test report that collects all existing differences.

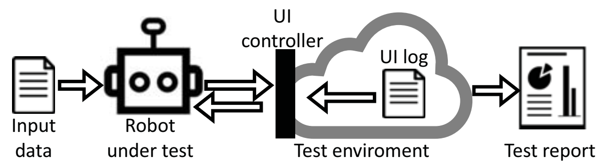


Fig. 7. Automatic testing process

Test nº: 1	Date: 2019/02/06	Duration: 12s.	Status: FAILED
Trace:	2019/02/16-0:8:20	*Keystroke "Tari Tavarez"*	
	2019/02/16-0:8:32	***Failure***	
Cause:	Expected	*Click: left, 120, 205*	
	but received:	*Keystroke "Burton Bertran"*	

Fig. 8. Example of a test report

Algorithm 1: Test Execution

```
input  : UILog $l$, UIController $c$
output: TestReport
1 $succEvents \leftarrow \emptyset$
2 foreach $expEvent : l$ do
3     $c$.showView($expEvent$.getImage())
4     $recEvent \leftarrow c$.waitForEvent()
5     if similarEvent($expEvent$,$recEvent$) then
6         $succEvents$.add($recEvent$)
7     else
8         return
          createFailReport($succEvents$,$expEvent$,$recEvent$)
9 return createPassReport($succEvents$)
```

The algorithm star with an empty set of successfully performed events (cf. line 1), and then goes through the events in the UI log (cf. line 2) and requires the UIController to show the image corresponding to the current event (cf. line 3).

Example 2: Regarding the UI log of Figure 6, *img_12* is shown first in the automatic testing since it is the image of the first event.

The algorithm then waits until the robot performs any action[2] and, after that, the received event is stored (cf. line 4) and it is verified whether it is similar to the current event in the UI log (cf. line 5). In the case when the events are similar (e.g., clicking on similar regions or introducing the same keystrokes), the received event is stored in the set of successful events (cf. line 6) and the algorithm continues with the subsequent event in the UI log. In turn, if the events diverge, a *fail report* is created, which details: (1) which events have been successfully performed (i.e., *succEvents*), (2) which events have failed (i.e., *recEvent*); and (3) which events were expected (i.e., *expEvent*, cf. line 8).

Example 3: Regarding the running example, the first expected event is a text action containing "Tari Tavarez". Assuming correct behaviour of the robot, the test environment will show the image of the subsequent event (i.e., *img_13*) and wait for an event of the robot. Assuming the robot introduces the text "Burton Bertman, the test will then be completed by producing the failure report of Figure 8.

The algorithm continues until all the expected events of the UI log are processed. On completion of the processing, a *pass report* is produced using the data stored in *succEvents* (cf. line 9).

When the algorithm ends, one of two cases may occur:

1) If all the events are validated, it means that the robot has done all its work correctly and this fact can be seen in the pass report. At the same time, it means that the robot is ready to be deployed in the real environment and hence, the lifecycle may continue.
2) If any event stops the process because it fails to match the expected events, then a fail report is returned stating that the robot is not ready for deployment, and hence the robot must be repaired before continuing the lifecycle.

VI. Conclusions and Future Work

Throughout this paper, a proposal to automatically test RPA projects has been presented. To this end, a test environment, which simulates the behaviour of the real system, is constructed based on UI log information. Robots are then tested against this synthetic system instead of the real system, thus reducing the risk of damaging a real system. This proposal is currently being validated in a real R&D project in the company Servinfom S.A. funded by the Spanish government. Thanks to this validation, a prototype could be developed and the preliminary results obtained seem promising.

Nonetheless, during the development of this proposal, certain limitations have been detected that could open new research lines. First, this paper presents a strict testing methodology, in that it depends directly on the behaviour of the user performed against the IS. It is important to emphasize that in case that additional environments (i.e. device configuration, specific software versions, etc.) or test cases want to be generated, such environments and behaviours must be generated.

Therefore, there is only one way to execute the operation although, in fact, it can be done in many ways. For instance, moving down on a component can be done by scrolling over that component or clicking on the "down arrow" till finding the desired element, or filling in a form can be carried out in orders without affecting the result. Second, this proposal is based on the fact that an UI log exists. What is more, this log is considered to contain information about the execution of a set of processes by a human. Although this is a strong assumption, this kind of logs are not so unusual in the BPO context. Finally, the generated test environment comprises a single long test covering the full UI log which is provided. However, such a log may contain several instances of different cases, e.g., the running example process presented in this paper

[2]The *waitForEvent* function may stop the algorithm if it waits too long for a robot interaction (e.g., if the robot freezes).

(cf. Figure 1) has two cases: the student already has a grade or the student has no grade. Despite the fact that the execution of a test for each case would suffice, our proposal tests the whole log which may invest unnecessary time. What is more, only one report is generated, which fails to differentiate between the cases.

As a consequence of the limitations identified, certain interesting research lines are opening up.

1) One of the most significant lines is the inference of the behaviour of the UI components to make the test more flexible. This can be carried out, for example, using machine learning techniques. This arrangement will allow the components of the windows to be treated as black boxes, by focusing on the result which is obtained after its execution and not on the behaviour at a mouse-click level.
2) In order to improve the efficiency of the test execution, the log needs to be pre-processed and treated so that it can be separated into separate cases (e.g., in the running example, one case can be each student). Although it lies outside the context of this paper, the process-mining paradigm positions itself as a firm candidate for the provision of a solution to this problem. Hence, instead of considering one single long test case, a test suite can be generated with a reduced number of shorter test cases which are necessary to cover the possible types of behaviours of the system to be simulated.
3) It is necessary to perform an empirical validation with a variety of contexts. Although, as mentioned above, this proposal is currently being validated in a real project, on the one hand, its results must be reported and, on the other hand, other case studies would be necessary in order to further corroborate the results in a more reliable way.

ACKNOWLEDGEMENTS

This research has been supported by the Pololas project (TIN2016-76956-C3-2-R) of the Spanish Ministry of Economy and Competitiveness and the RAIL project (P114-16/E09) of the Centre for Industrial Technological Development (Centro para el Desarrollo Tecnológico Industrial, CDTI) of Spain.

REFERENCES

[1] S. Aguirre and A. Rodriguez, "Automation of a business process using robotic process automation (rpa): A case study," in *Workshop on Engineering Applications*. Springer, 2017, pp. 65–71.
[2] S. Anagnoste, "Robotic automation process - the next major revolution in terms of back office operations improvement," *Proceedings of the International Conference on Business Excellence*, vol. 11, 07 2017.
[3] A. Asatiani and E. Penttinen, "Turning robotic process automation into commercial success–case opuscapita," *Journal of Information Technology Teaching Cases*, vol. 6, no. 2, pp. 67–74, 2016.
[4] G. Barnett, "Robotic process automation: Adding to the process transformation toolkit," *White paper IT0022-0005, Ovum Consulting*, 2015.
[5] Blue Prism, "www.blueprism.com," [Online; accessed January 2019].
[6] H. P. Fung, "Criteria, use cases and effects of information technology process automation (itpa)," *Advances in Robotic and Automation*, no. 3, pp. 1–11, 2014.
[7] J. Geyer-Klingeberg, J. Nakladal, F. Baldauf, and F. Veit, "Process mining and robotic process automation: A perfect match," in *International Conference on Business Process Management*, 2018, pp. 1–8.
[8] J. Hultin, C. Trudell, A. Vashistha, and T. Glover, "Implications of technology on the future workforce," Defense Business Board Washington United States, Tech. Rep., 2017.
[9] T. Kämäräinen *et al.*, "Managing robotic process automation: Opportunities and challenges associated with a federated governance model," Master's thesis, School of Business, 2018.
[10] M. Lacity and L. Willcocks, "Robotic process automation: the next transformation lever for shared services," *London School of Economics Outsourcing Unit Working Papers*, vol. 7, 2015.
[11] C. Lamberton, D. Brigo, and D. Hoy, "Impact of robotics, rpa and ai on the insurance industry: challenges and opportunities," *Journal of Financial Perspectives: Insurance edition*, 2017.
[12] C. Le Clair, A. Cullen, and M. King, "The forrester wave: Robotic process automation, q1 2017," 2017.
[13] V. Leno, M. Dumas, F. M. Maggi, and M. La Rosa, "Multi-perspective process model discovery for robotic process automation," *CEUR Workshop Proceedings*, vol. 2114, pp. 37–45, 2018.
[14] H. Leopold, H. van der Aa, and H. A. Reijers, "Identifying candidate tasks for robotic process automation in textual process descriptions," in *Enterprise, Business-Process and Information Systems Modeling*. Springer, 2018, pp. 67–81.
[15] C. Linn, P. Zimmermann, and D. Werth, "Desktop activity mining-a new level of detail in mining business processes," in *Workshops der INFORMATIK 2018-Architekturen, Prozesse, Sicherheit und Nachhaltigkeit*. Köllen Druck+ Verlag GmbH, 2018.
[16] H. Lu, Y. Li, M. Chen, H. Kim, and S. Serikawa, "Brain intelligence: go beyond artificial intelligence," *Mobile Networks and Applications*, vol. 23, no. 2, pp. 368–375, 2018.
[17] S. Madakam, R. M. Holmukhe, and D. K. Jaiswal, "The Future Digital Work Force: Robotic Process Automation (RPA)," *JISTEM - Journal of Information Systems and Technology Management*, vol. 16, 00 2019.
[18] P. Mijović, E. Giagloglou, P. Todorović, I. Mačužić, B. Jeremić, and I. Gligorijević, "A tool for neuroergonomic study of repetitive operational tasks," in *Proceedings of the 2014 European Conference on Cognitive Ergonomics*. ACM, 2014, p. 32.
[19] V. K. Naik, P. Garbacki, and A. Mohindra, "Architecture for service request driven solution delivery using grid systems," in *Services Computing, 2006. SCC'06. IEEE International Conference on*. IEEE, 2006, pp. 414–422.
[20] E. Penttinen, H. Kasslin, and A. Asatiani, "How to choose between robotic process automation and back-end system automation?" in *European Conference on Information Systems*, 2018.
[21] J. R. Slaby, "Robotic Automation emerges as a threat to traditional low-cost outsourcing," https://www.blueprism.com/wpapers/robotic-automation-emerges-threat-traditional-low-cost-outsourcing, 2018, [Online; accessed January 2019].
[22] UiPath, "www.uipath.com," [Online; accessed January 2019].
[23] W. M. van der Aalst, M. Bichler, and A. Heinzl, "Robotic process automation," 2018.
[24] L. Willcocks, M. Lacity, and A. Craig, "Robotic process automation: strategic transformation lever for global business services?" *Journal of Information Technology Teaching Cases*, vol. 7, no. 1, pp. 17–28, 2017.
[25] WorkFusion RPA Express, "www.workfusion.com/rpaexpress," [Online; accessed January 2019].