Short Paper

Intelligent Robotic Process Automation: Generating Executable RPA Scripts from Unsegmented UI Logs

Simone Agostinelli[1], Andrea Marrella[1]

[1]*Sapienza Università di Roma - Dipartimento di Ingegneria Informatica Automatica e Gestionale Antonio Ruberti*

Abstract

Robotic Process Automation (RPA) is an automation technology that sits between the fields of Business Process Management (BPM) and Artificial Intelligence (AI) that creates software (SW) robots to partially or fully automate rule-based and repetitive tasks (or simply routines) performed by human users in their applications' user interfaces (UIs). RPA tools are able to capture in dedicated UI logs the execution of many routines of interest. A UI log consists of user actions that are mixed in some order that reflects the particular order of their execution by the user, thus potentially belonging to different routines. When considering state-of-the-art RPA technology in the BPM domain, it becomes apparent that the current generation of RPA tools is driven by predefined rules and manual configurations made by expert users rather than intelligent techniques. In this paper, we discuss our research targeted at injecting intelligence into RPA practices. Specifically, we present an approach to: *(i)* automatically understand which user actions contribute to which routines inside a UI log (this issue is known as segmentation) and *(ii)* automatically generate executable RPA scripts directly from the UI logs that record the user interactions with the SW applications involved in a routine execution, thus skipping completely the (manual) modeling activity of the flowchart diagrams.

1. Introduction

Robotic Process Automation (RPA) is an emerging automation technology in the Business Process Management (BPM) domain [1] that creates software (SW) robots to partially or fully automate rule-based and repetitive tasks (or simply *routines*) performed by human users in their applications' user interfaces (UIs) [2]. While conducting a BPM project is often considered too expensive because its "top-down" approach, RPA relies on an approach where, rather than redesigning existing information systems (that remain unchanged), humans are replaced by SW robots in the execution of those business processes (BPs) involving routine work.

In the research literature, many case studies have shown that RPA technology can concretely lead to improvements in efficiency for BPs involving routine work in large companies (e.g., Vodafone) [3]. Moreover, the market of RPA solutions has developed rapidly. Today includes more than 50 vendors developing tools that provide SW robots with advanced functionalities for automating office tasks of different complexity [4].

However, despite the growing attention around RPA, to achieve a widespread adoption in the BPM domain, RPA needs to become "smarter" [5, 6, 7], i.e., RPA tools should employ AI

PMAI@IJCAI 2022 (International Workshop on Process Management in the AI era), July 23–25, 2022, Vienna, Austria
agostinelli@diag.uniroma1.it (S. Agostinelli); marrella@diag.uniroma1.it (A. Marrella)

solutions to adapt and learn how to handle non-standard cases by observing human resolving unexpected system behaviour (e.g., in case of system errors, etc.). In this direction, we discuss in this paper three relevant research challenges in the RPA field and we present an approach to increase their automated features making them less dependent by direct human involvement.

2. Background

The traditional workflow to conduct an RPA project can be summarized as follows [8]: (1) `determine` which process steps are good candidates to be automated; (2) `model` the selected routines in the form of *flowchart diagrams*, which involve the specification of the actions, routing constructs, data flow, etc. that define the behaviour of a SW robot; (3) `record` the mouse/key events that happen on the UI of the user's computer system; (4) `develop` each modeled routine by generating the SW code required to concretely enact the associated SW robot on a target computer system; (5) `deploy` the SW robots in their environment to perform their actions; and (6) `monitor` the performance of SW robots to detect bottlenecks and exceptions.

The majority of the previous steps, particularly the ones involved in the early stages of the RPA life-cycle, require the support of skilled human experts, which need to: *(i)* understand the anatomy of the candidate routines to automate through interviews, walk-troughs, and detailed observation of workers conducting their daily work (cf. step 1); and *(ii)* define manually the flowchart diagrams representing the structure of such routines (cf. step 2), which will drive the development of the SW code, often in the form of executable scripts (also called RPA scripts), allowing the concrete enactment of SW robots at run-time (cf. steps 3 and 4).

While this approach is effective to execute simple rules-based logic in situations where there is no room for interpretation, it becomes time-consuming and error-prone in presence of routines that are less predictable or require some level of human judgment. The issue is that in case where the flowchart diagram does not contain a suitable response for a specific situation, e.g., because of an inaccurate modeling activity, then the associated RPA scripts would not properly reflect the behaviour of the potential routine variant, forcing SW robots to escalate to a human supervisor at run-time, in contrast with the RPA philosophy.

3. Approach

Three major challenges arise from the above discussion [5, 9]: (**C1**) the automated identification of the routine steps to robotize from a UI log (cf., the *segmentation* issue [10]), (**C2**) the automated detection of all the routine variants that require some user input to proceed with their execution, and (**C3**) the automated synthesis of executable RPA scripts for enacting SW robots at run-time. To address these challenges, we have developed an approach and an implemented tool, called SmartRPA [11, 12], which is able to: *(i)* interpret the UI logs recording the mouse/key events that happen on the UI of the SW applications involved in many routine executions, *(ii)* automatically identify and understand which user actions contribute to a particular routine inside a UI log and cluster them into well-bounded routine traces, *(iii)* discover all the variants of the routine under observation, and *(iv)* automatically combine them into an executable RPA script, which can be reactively synthesized into a single SW robot.

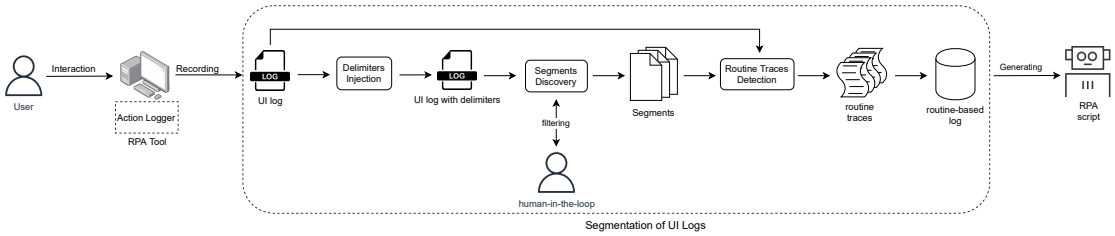

Figure 1: Overview of the adopted approach

To achieve these goals, as shown in Figure 1, starting from an unsegmented UI log previously recorded with an RPA tool, the first stage of this research is to inject into the UI log the *end-delimiters* of the routines under examination. An end-delimiter is a dummy action added to the UI log immediately after the user action that is known to complete a routine execution. The knowledge of such end-delimiters is crucial to make the approach works, as discussed in [13].

For tackling **C1**, we rely on three main steps: *(i)* a *frequent-pattern identification technique* [14] (customized on a ad-hoc basis) to automatically derive the routine segments from a UI log (i.e., *routine segments* describe the different behaviours of the routine(s) under analysis, in terms of repeated patterns of performed user actions), *(ii)* a *human-in-the-loop interaction* to filter out those segments not allowed (i.e., wrongly discovered from the UI log) by any real-world routine execution by means of *declarative constraints* [15], and *(iii)* a routine traces detection component that leverages *trace alignment* in Process Mining [16] to cluster all user actions belonging to a specific routine segment into well-bounded routine traces (i.e., a *routine trace* represents an execution instance of a routine within a UI log). Such traces are finally stored in a *routine-based log*, which captures all the user actions happened during many routine executions.

Commercial RPA tools can employ routine-based logs to synthesize executable scripts in the form of SW robots without the manual modeling of the routines (cf. steps 3 and 4). To this end, the SmartRPA approach[1] is able to automatically synthesize executable scripts for enacting SW robots at run-time. The SW robots generated by SmartRPA are obtained to handle the intermediate user inputs that are required during the routine execution, thus enabling to emulate the most suitable routine variant for any specific combination of user inputs as observed in the UI log (**C2**). This makes the synthesis of SW robots performed by SmartRPA approach *reactive* to any user decision found during a routine execution, thus allowing the potential run-time generation of as many SW robots as the routine variants to be emulated (**C3**).

4. Concluding Remarks

In summary, this research tries to mitigate the involvement of skilled human experts in steps 1, 2, 3, and 4, throughout the development of the SmartRPA approach which is able to tackle **C1**, **C2** and **C3**. The main weakness of the SmartRPA approach is correlated with the quality of information recorded in real-world UI logs. Since a UI log is fine-grained, routines executed

[1] https://github.com/bpm-diag/smartRPA

with many different strategies may potentially affect the identification of the routine segments.

In conclusion, our approach is based on a semi-supervised assumption, since we know a-priori the end-delimiters to be associated to any user action that ends a routine execution. On the other hand, the approach is not aware of the concrete behaviour of the routines of interest, which will be discovered by the approach itself. For this reason, we consider this contribution as an important step towards the development of a more complete and unsupervised approach. Towards this direction, we aim at relaxing the semi-supervised assumption by employing *machine learning* and *DNN* techniques to automatically identify the end-delimiters.

Acknowledgments. This work has been supported by the H2020 project DataCloud.

References

[1] M. Kirchmer, Robotic Process Automation-Pragmatic Solution or Dangerous Illusion, BTOES Insights, June'17 (2017).
[2] W. M. P. van der Aalst, Robotic Process Automation, Bus. Inf. Syst. Eng. 60 (2018).
[3] S. Aguirre, A. Rodriguez, Automation of a Business Process Using Robotic Process Automation (RPA): A Case Study, in: Appl. Comp. Scien. in Eng., Springer, 2017, pp. 65–71.
[4] AI-Multiple, Top 53 RPA Tools, 2022. URL: https://blog.aimultiple.com/rpa-tools/.
[5] S. Agostinelli, A. Marrella, M. Mecella, Research Challenges for Intelligent Robotic Process Automation, in: Business Process Management Workshops, 2019, pp. 12–18.
[6] T. Chakraborti, et al., From Robotic Process Automation to Intelligent Process Automation: Emerging Trends, in: RPA Forum '20, Springer, 2020, pp. 215–228.
[7] S. Agostinelli, A. Marrella, M. Mecella, Towards Intelligent Robotic Process Automation for BPMers. arXiv:2001.00804, 2020.
[8] A. Jimenez-Ramirez, H. A. Reijers, I. Barba, C. Del Valle, A Method to Improve the Early Stages of the Robotic Process Automation Lifecycle, in: CAiSE'19, 2019, pp. 446–461.
[9] V. Leno, A. Polyvyanyy, M. Dumas, M. La Rosa, F. M. Maggi, Robotic Process Mining: Vision and Challenges, Bus. Inf. Syst. Eng. (2020) 1–14.
[10] S. Agostinelli, A. Marrella, M. Mecella, Automated Segmentation of User Interface Logs. In: RPA. Management, Technology, Applications, De Gruyter, 2021.
[11] S. Agostinelli, M. Lupia, A. Marrella, M. Mecella, Automated Generation of Executable RPA Scripts from User Interface Logs, in: RPA Forum '20, volume 393, 2020, pp. 116–131.
[12] S. Agostinelli, M. Lupia, A. Marrella, M. Mecella, Reactive synthesis of software robots in RPA from user interface logs, Comp. in Industry 142 (2022).
[13] S. Agostinelli, F. Leotta, A. Marrella, Interactive Segmentation of User Interface Logs, in: 19th Int. Conf. on Service-Oriented Computing (ICSOC), volume 13121, 2021, pp. 65–80.
[14] D. J. Cook, N. C. Krishnan, P. Rashidi, Activity Discovery and Activity Recognition: A New Partnership, IEEE Transactions on Cybernetics 43 (2013) 820–828.
[15] W. M. van Der Aalst, M. Pesic, H. Schonenberg, Declarative Workflows: Balancing between Flexibility and Support, Comp. Sc.-Res. and Dev. 23 (2009).
[16] M. de Leoni, A. Marrella, Aligning Real Process Executions and Prescriptive Process Models through Automated Planning, Expert System with Application 82 (2017) 162–183.