Mastering Robotic Process Automation with Process Mining

Simone Agostinelli[1], Andrea Marrella[1], Luka Abb[2], and Jana-Rebecca Rehse[2]

[1] Sapienza Universitá di Roma, Rome, Italy
{agostinelli,marrella}@diag.uniroma1.it
[2] Universität Mannheim, Mannheim, Germany
{luka.abb,rehse}@uni-mannheim.de

Abstract. Robotic Process Automation (RPA) is an emerging automation technology that creates software (SW) robots to partially or fully automate rule-based and repetitive tasks (aka routines) previously performed by human users in their applications' user interfaces (UIs). Successful usage of RPA requires strong support by skilled human experts, from the detection of the routines to be automated to the development of the executable scripts required to enact SW robots. In this paper, we discuss how process mining can be leveraged to minimize the manual and time-consuming steps required for the creation of SW robots, enabling new levels of automation and support for RPA. We first present a reference data model that can be used for a standardized specification of UI logs recording the interactions between workers and SW applications to enable interoperability among different tools. Then, we introduce a pipeline of processing steps that enable us to (1) semi-automatically discover the anatomy of a routine directly from the UI logs, and (2) automatically develop executable scripts for performing SW robots at run-time. We show how this pipeline can be effectively enacted by researchers/practitioners through the SmartRPA tool.

Keywords: Robotic Process Automation · Process Mining · User Interface (UI) Logs · Reference Data Model for UI logs · Segmentation · Automated Generation of SW Robots from UI Logs · SmartRPA

1 Introduction

Robotic Process Automation (RPA) is an emerging automation technology in the Business Process Management (BPM) domain 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) [1]. Despite the growing attention around RPA, when considering state-of-the-art RPA technology, it becomes apparent that the current generation of RPA tools is driven by predefined rules and manual configurations made by expert users rather than automated techniques [7,8,10].

The traditional life-cycle of an RPA project can be summarized as follows [14]: (1) `determine` which process steps are good candidates to be automated in

Pre-print copy of the manuscript published by Springer
(available at: https://link.springer.com/chapter/10.1007/978-3-031-16103-2_6)
identified by doi: 10.1007/978-3-031-16103-2_6

the form of routines; (2) `model` the selected routines trough *flowchart diagrams*, which involve the specification of the actions, routing constructs, data flow, etc. that define the behaviour of a SW robot; (3) `develop` each modeled routine by generating the SW code required to concretely enact the associated SW robot on a target computer system; (4) `deploy` the SW robots in their environment to perform their actions; (5) `monitor` the performance of SW robots to detect bottlenecks and exceptions; and (6) `maintain` routines over time, updating the SW robots when needed. The majority of the previous steps, particularly the early ones, require the support of skilled human experts, which need to 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 manually define the flowchart diagrams representing the structure of such routines, cf. step (2). These diagrams will drive the development of the executable scripts (also called RPA scripts), allowing for the concrete enactment of SW robots at run-time, cf. steps (3) and (4). The problem is that this high degree of human involvement contradicts the underlying objective of RPA, i.e., an increased level of automation.

In this paper, we discuss how process mining can be leveraged to address this problem, enabling new levels of automation and support for RPA. Building on the RPM (Robotic Process Mining) framework [16], we show that the generation of SW robots can be achieved in a semi-automated way directly from the UI logs recording the interactions between workers and SW applications during one or more routine(s) executions, thus eliminating the manual and time-consuming steps (1) and (2) required for modeling the details of the routine structure.

Specifically, in Section 2, we first present a reference data model that enables a standardized specification of UI logs. Then, in Section 3, we show how the RPM framework can be effectively enacted by researchers/practitioners through the SmartRPA approach [4,6] and its implemented tool [5], which enables to interpret the UI logs keeping track of many routine executions, and to generate SW robots that emulate the most suitable routine variant for any specific intermediate user input that is required during the routine execution. Finally, in Section 4 we conclude the paper by tracing future work.

2 Specifying and Collecting UI Logs

The main source of data for RPA are UI logs, which are a particular kind of event log that record low-level manual activities during the execution of a task in an information system. Examples of events in a UI log include clicking a button, entering a string into a text field, ticking a checkbox, or selecting a value from a dropdown. The specific scope of a UI log, including the definition of relevant activities and attributes to cover, depends on the context in which the log is collected and the purpose for which it is used. Hence, the first challenges when collecting UI logs are often (1) to determine what kind of data is available and (2) to design the data collection process so that the logs are comprehensive enough to cover the desired automation use cases.

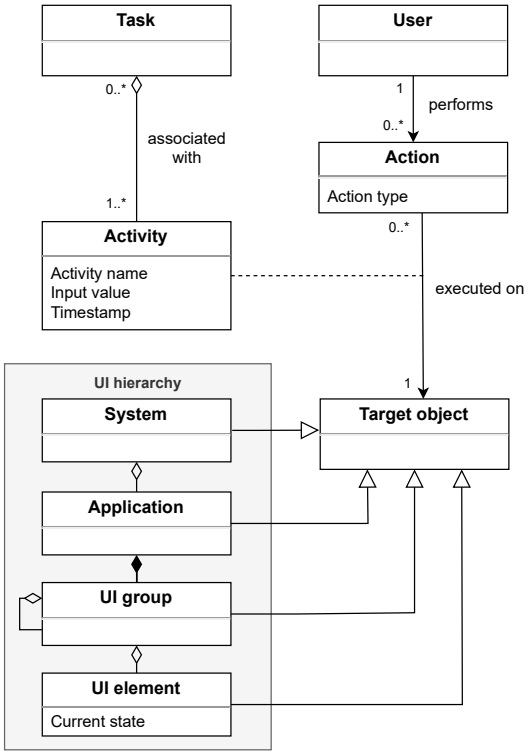

Fig. 1. Reference data model for UI logs [2]

To specify a UI log for RPA, one needs to determine which attributes can and should be recorded and how they relate to each other. The UI log should be as standardized as possible to allow for interoperability between different tools, but they also need to be adapted to the individual scenario. To achieve this, they can refer to the reference data model for process-related UI logs, shown in Fig. 1. This reference model defines the core attributes of UI logs but remains flexible with regard to the scope, level of abstraction, and case notion [2]. It defines the activity of a UI log as a combination of an action (e.g., *click* or *input*) and a target object in the user interface. It further specifies the possible instances of target objects and their hierarchical relation, as well as task and user components that provide additional (business) context.

After specifying the UI log structure, the actual data needs to be recorded. Generally, there are three ways to achieve this: application-independent logging with screen capture and OCR technology [18,14], application-specific logging with plug-ins [4,17], and application-internal logging within the an application's source code. Not all options are feasible in each application context and they each have certain assets and drawbacks. For example, application-internal logging will typically produce the highest data quality, but it is only possible if we have access to the application's source code. Application-independent and application-specific logging have to externally reconstruct the events that happen within the application, but can be applied to any tool independent of its origin.

3 SmartRPA: From UI logs to SW robots

The approach underlying SmartRPA takes inspiration from the RPM framework presented by Leno et al. in [16]. RPM aims to support analysts to produce

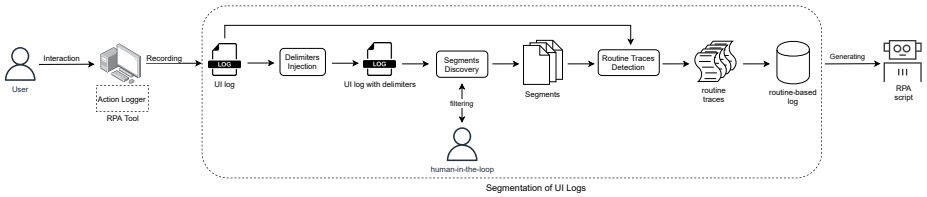

Fig. 2. Overview of the SmartRPA approach

executable specifications of routines, in form of SW robots, interpreting the routine executions stored in a UI log. Specifically, RPM envisions a pipeline of two main stages that consist of: *(i)* interpreting UI logs corresponding to executions of one or more routine executions, by identifying the candidate routines to be automated with RPA tools (i.e., the *segmentation* issue [9]); and *(ii)* synthesizing executable RPA scripts to enact SW robots. SmartRPA incorporates these stages within a larger approach, as shown in Fig. 2.

Starting from an unsegmented UI log previously recorded with an RPA tool, the first stage of the SmartRPA approach 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 work, as discussed in [3].

For tackling the segmentation issue, we rely on three main steps: *(i)* a *frequent-pattern identification technique* [11] 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* [13], and *(iii)* a routine traces detection component that leverages *trace alignment* in process mining [12] 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 dedicated *routine-based log*, which captures exactly all the user actions happened during many different executions of the routine.

Commercial RPA tools can eventually employ routine-based logs to synthesize executable scripts in the form of SW robots that will emulate the routine behaviour on the UI without the manual modeling of the routines. To this end, the SmartRPA tool[3] is able to automatically synthesize executable scripts for enacting SW robots at run-time. Notably, 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. This

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

makes the synthesis of SW robots performed by SmartRPA *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 [6].

4 Concluding Remarks

The goal of RPA is to automate routines and high-volume tasks, but it currently requires substantial manual intervention of expert users. In this paper, we offer a twofold contribution towards an intelligent and fully automated generation of SW robots from the users' observed behavior as recorded in UI logs. First, we introduce a reference data model for a standardized specification of UI logs, which enforces interoperability among different RPM-based tools. Second, we present a pipeline of processing steps, implemented trough the SmartRPA approach, to develop executable RPA scripts by solely interpreting the UI logs at hand.

The reference model provides a common, application-independent conceptual framework for user interactions. However, it still has to prove its utility in practice. We therefore want to encourage readers to adopt the model for capturing UI logs in their projects. Compared with the literature approaches to automated RPA script generation from UI logs [15,18], which enable to automate only the most frequent routine variant among the ones discovered in the UI log, SmartRPA provides a reactive approach that emulates the most suitable routine variant for any specific intermediate user input that is required during the routine execution. As a consequence, this makes the working of SW robots generated by SmartRPA flexible and adaptable to several real-world situations.

The main weakness of SmartRPA relates to the quality of information recorded in real-world UI logs. Since a UI log is fine-grained, routines executed with many different strategies may potentially affect the identification of the routine segments. In addition, SmartRPA is based on a semi-supervised assumption, since the end-delimiters required to untangle the segmentation issue are known a-priori. Conversely, on the positive side, the employed segmentation technique is able to outperform existing literature approaches in terms of supported segmentation variants, in particular when there are many interleaved routine executions recorded in the UI log [3]. For this reason, we consider this contribution as an important step towards the development of an unsupervised approach that employs machine learning techniques to automatically identify the end-delimiters.

Acknowledgments. This work has been partially supported by the the H2020 project DataCloud and the Sapienza grant BPbots.

References

1. van der Aalst, W.M.P., Bichler, M., Heinzl, A.: Robotic Process Automation. Bus. Inf. Syst. Eng. **60**(4), 269–272 (2018). https://doi.org/10.1007/s12599-018-0542-4
2. Abb, L., Rehse, J.R.: A Reference Data Model for Process-Related User Interaction Logs. In: Int. Conf. on Business Process Management (BPM'22) (2022)

3. Agostinelli, S., Leotta, F., Marrella, A.: Interactive Segmentation of User Interface Logs. In: Int. Conf. on Service-Oriented Computing (ICSOC'21). pp. 65–80 (2021). https://doi.org/10.1007/978-3-030-91431-8_5
4. Agostinelli, S., Lupia, M., Marrella, A., Mecella, M.: Automated Generation of Executable RPA Scripts from User Interface Logs. In: RPA Forum. pp. 116–131 (2020). https://doi.org/10.1007/978-3-030-58779-6_8
5. Agostinelli, S., Lupia, M., Marrella, A., Mecella, M.: SmartRPA: A Tool to Reactively Synthesize Software Robots from UI Logs. In: CAiSE Forum. pp. 137–145 (2021). https://doi.org/10.1007/978-3-030-79108-7_16
6. Agostinelli, S., Lupia, M., Marrella, A., Mecella, M.: Reactive Synthesis of Software Robots in RPA from User Interface Logs. Computers in Industry (2022)
7. Agostinelli, S., Marrella, A., Mecella, M.: Research Challenges for Intelligent Robotic Process Automation. In: BPM Workshops. pp. 12–18 (2019). https://doi.org/10.1007/978-3-030-37453-2_2
8. Agostinelli, S., Marrella, A., Mecella, M.: Towards Intelligent Robotic Process Automation for BPMers (2020), http://arxiv.org/abs/2001.00804
9. Agostinelli, S., Marrella, A., Mecella, M.: Exploring the Challenge of Automated Segmentation in Robotic Process Automation. In: Int. Conf. on Research Challenges in Information Science (2021). https://doi.org/10.1007/978-3-030-75018-3_3
10. Chakraborti, T., Isahagian, V., Khalaf, R., Khazaeni, Y., Muthusamy, V., Rizk, Y., Unuvar, M.: From Robotic Process Automation to Intelligent Process Automation: Emerging Trends. In: Blockchain and RPA Forum. pp. 215–228. Springer (2020). https://doi.org/10.1007/978-3-030-58779-6_15
11. Cook, D.J., Krishnan, N.C., Rashidi, P.: Activity Discovery and Activity Recognition: A New Partnership. IEEE Transactions on Cybernetics **43**(3), 820–828 (2013). https://doi.org/10.1109/TSMCB.2012.2216873
12. de Leoni, M., Lanciano, G., Marrella, A.: Aligning Partially-Ordered Process-Execution Traces and Models Using Automated Planning. In: Twenty-Eight Int. Conf. on Automated Planning and Scheduling (ICAPS 2018). pp. 321–329 (2018), https://aaai.org/ocs/index.php/ICAPS/ICAPS18/paper/view/17739
13. van Der Aalst, W.M., Pesic, M., Schonenberg, H.: Declarative Workflows: Balancing between Flexibility and Support. Comp. Sc.-Res. and Dev. **23**(2) (2009). https://doi.org/10.1007/s00450-009-0057-9
14. Jimenez-Ramirez, A., Reijers, H.A., Barba, I., Del Valle, C.: A Method to Improve the Early Stages of the Robotic Process Automation Lifecycle. In: Int. Conf. on Advanced Information Systems Engineering (CAiSE'19). pp. 446–461 (2019). https://doi.org/10.1007/978-3-030-21290-2_28
15. Leno, V., Deviatykh, S., Polyvyanyy, A., Rosa, M.L., Dumas, M., Maggi, F.M.: Robidium: Automated Synthesis of Robotic Process Automation Scripts from UI Logs. In: BPM Demonstration & Resources (2020), http://ceur-ws.org/Vol-2673/paperDR08.pdf
16. Leno, V., Polyvyanyy, A., Dumas, M., La Rosa, M., Maggi, F.M.: Robotic Process Mining: Vision and Challenges. Bus. Inf. Syst. Eng. pp. 1–14 (2020). https://doi.org/10.1007/s12599-020-00641-4
17. Leno, V., Polyvyanyy, A., Rosa, M.L., Dumas, M., Maggi, F.M.: Action Logger: Enabling Process Mining for Robotic Process Automation. In: BPM Demonstration & Resources (2019), http://ceur-ws.org/Vol-2420/paperDT2.pdf
18. Linn, C., Zimmermann, P., Werth, D.: Activity Mining - A new level of detail in mining business processes. In: Workshops der INFORMATIK. pp. 245–258 (2018), https://dl.gi.de/20.500.12116/17225