Towards an OpenSource Logger for the Analysis of RPA Projects

José Manuel López-Carnicer, Carmelo del Valle, and José González Enríquez(✉)

Computer Languages and Systems Department, Escuela Técnica Superior de Ingeniería Informática, Avenida Reina Mercedes, s/n, 41012 Sevilla, Spain
joslopcar@alum.us.es, {carmelo,jgenriquez}@us.es

Abstract. Process automation typically begins with the observation of humans conducting the tasks that will be eventually automated. Similarly, successful RPA projects require a prior analysis of the undergoing processes which are being executed by humans. The process of collecting this type of information is known as user interface (UI) logging since it records the interaction against a UI. Main RPA platforms (e.g., Blueprism and UIPath) incorporate functionalities that allow the recording of these UI interactions. However, the records that these platforms generate lack some functionalities that large-scale RPA projects require. Besides, they are only understandable by the proper RPA platforms. This paper presents an extensible and multi-platform OpenSource UI logger that generate UI logs in a standard format. This system collects information from all the computers it is running on and sends it to a central server for its processing. Treatment of the collected information will allow the creation of an enriched UI log which can be used, among others purposes, for smart process analysis, machine learning training, the creation of RPA robots, or, being more general, for task mining .

Keywords: RPA · Computer-human interaction · OpenSource project · Process discovery · Task mining

1 Introduction

The emerging technology of Robotic Process Automation (RPA) is said to enable the automation of the most repetitive, tedious, and mundane digital tasks that people are suing to do [1,25]. However, not every task is suitable to be robotised since, besides these characteristics, it should (1) have a low level of exceptions, (2) require an enclosed cognitive effort, and (3) be susceptible to human errors [8]. Therefore, successful RPA projects require to start with an analysis phase [7] which identify those candidate processes—or part of them—which have more chances to be robotised in a cost-effective way, i.e., those which guarantees the highest return of the inversion with the lowest risk. Although most of the time this analysis mainly rely on interpreting process documentation, the latter may

A. Asatiani et al. (Eds.): BPM Blockchain and RPA Forum 2020, LNBIP 393, pp. 176–184, 2020.
https://doi.org/10.1007/978-3-030-58779-6_12

be of poor quality and may require substantial effort to understand [11]. In consequence, there is an increasing trend to capture the actual behaviour of the people interacting with real information systems (ISs) to amend the documentation problems, i.e., recording interaction events like mouse clicks or keystrokes.

Both academia and industry have acknowledged this issue and provide a variety of approaches. On the one hand, vendor-specific platforms (e.g., BluePrism [5], AutomationAnywhere [2], and UIPath [22]) offer tools to record macros-like scripts from the computer of a user executing the process tasks [23]. The obtained script can be analyzed later through the own vendor platform to discover the candidate process and, even, to support the robot code development. On the other hand, proposals can be found in the literature that suggest the creation of a standard log of events related to the interaction of the user with the graphical user interface, the so-called UI Log [6,11,12]. Obtaining this kind of log enables using the Process Mining paradigm [24] to disclose the knowledge that the log contains, among other things, the candidate processes to robotize. These proposals fall under the paradigm of Task Mining [16].

Although these solutions are reasonably mature, they lack support for real-world problems like those existing in the Business Process Outsourcing (BPO) industry, which presents one of the most suitable settings for conducting successful RPA projects [9]. The back-office of BPO departments is composed by large teams of workers performing digital processes through ISs of external companies. Creating a UI Log that comprises the behaviour of the whole team is a challenge for a series of reasons. Firstly, the distributed logging must be centralized in a common event log whose size increases with the size of the team. Secondly, the log of each member of the team may have some differences since not all the team share the same environment, e.g., screen resolution, text editor, WEB browser, etc. And, finally, each member of the team may perform the same processes differently than their teammates. Nowadays, this kind of distributed logging is not supported and, solutions that can be extended in this direction are not intended to be used in RPA, i.e., the generated log lacks detailed information for a thorough RPA analysis.

This paper motivates the minimum requirements that a UI logger should have in a distributed context based on an industrial collaboration with a company belonging to the BPO sector. In addition, software design is proposed to develop this logger as an Open Source project aiming to enable researchers and practitioners to easily get into RPA.

The rest of the paper is organized as follows. Section 2 describes the BPO context and identifies the fundamental requirements of this proposal. Section 3 describes a classification scheme where the most important features of the above requirements, and the tools related to this proposal, are categorized. Section 4 describes the proposed solution provided in this proposal. Finally, Sect. 5 summarises the work and presents some future work.

2 BPO Context

This section describes the knowledge flow which drives the interaction between all the participants in a BPO context which plan to acquire RPA capabilities.

In the context of the back-office employees, different training sessions provide them with the knowledge to perform an outsourced process against some ISs through their own computers. Different challenges are faced during the execution of such process since the real systems tend to present slight differences from the one learnt during the training, or even completely new process whose behavior is similar and they can adapt themselves to accomplish it. In addition, other issues may arise out of the processes like networking problems, operating system errors, etc. These challenges are typically addressed using their common sense and sharing the newly acquired knowledge with the rest of the back-office team.

In the context of the RPA analyst, similar training lessons are provided along with detailed documentation which is typically delivered by the company which host the process to outsource. This information is thoroughly analysed to (1) to understand the workflow and depict the *as-is* process, (2) to identify which parts of this process would be a good candidate to robotise, and (3) to provide a design of the robotised process to continue the RPA development. In this path, the RPA analyst recognises that there are chances that the prescribed process is not fully aligned with the real process. For this, this analyst uses to have periodic and informal interviews with some back-office employees to contrast to assimilate the on-the-field knowledge of them. Nonetheless, most of the details remain undisclosed within the back-office know-how.

For all these reasons, undesired effects occur from both perspectives, the RPA analyst and the back-office employees. RPA projects start with a higher uncertainty after long analysis periods and employees are sued to do mundane tasks for longer and unpredictable periods. For this, RPA analyst must be supported with a formal way to capture the back-office knowledge and which does not require intensive efforts from the back-office employees. Behavioural loggers present a suitable candidate that would be highly welcome by RPA analyst or the back-office employees.

After analysing this context, the following advanced requirements have been identified that are not typically offered by common loggers and that are the motivation of this paper: (1) The scalability level, i.e. the number of computers that can be monitored simultaneously, depending on the execution context, without impacting the system; (2) the method of sending the captured information to the user; (3) the facility to data processing, either through a database or some data structure that can be processed (log) and (4), the possibility of editing or complementing the features offered by the tool with the new software.

3 Related Work

Nowadays, different proposals provide the possibility of creating logs recording the behavior of a human interacting against a computer. In this sense, this section

Table 1. Classification scheme

	Keyboard	Mouse clicks	Mouse position	Clipboard	Screenshot	Foreground application	Capture moment	Specification	Multiplatform	Open source	Server	Remote control	Data access	Totals
Spyrix [18]	1	1	0	1	1	1	1	1	1	0	1	1	0.5	**10.5**
Spytech SpyAgent [19]	1	1	1	1	1	1	1	1	0	0	1	1	1	**11.0**
Action logger [13]	1	0	0	1	1	1	1	0	1	0	1	1	0.5	**8.0**
Best free keylogger [4]	1	0	0	1	1	1	1	0	0	0	1	0	1	**7.0**
Black cat [15]	1	1	0	0	1	1	1	0	0	1	1	0	1	**8.0**
UIPath [22]	1	1	1	1	0.5	1	0	1	1	0	0	0	0	**7.5**
OpenRPA [14]	1	1	1	1	0.5	1	0	1	1	1	0	0	0	**8.5**
Taskt [3]	1	1	1	1	0	1	0	0	1	1	0	0	0	**7.0**
Totals	**8.0**	**6.0**	**4.0**	**7.0**	**6.0**	**8.0**	**5.0**	**4.0**	**5.0**	**3.0**	**5.0**	**3.0**	**4.0**	

aims to describe the state-of-the-art regarding this topic, listing and categorizing the proposals found into a classification scheme.

In the context of parental and company employees control, many keylogger tools offer solutions for monitoring the activity of their users [4,15,18,19]. In addition, platforms with broader objectives, such as the creation and management of RPA projects, also offer users the possibility of recording their activity [3,20,21]. However, the generated logs are frequently understandable only in the context of the platform itself. The closest solution to the proposal presented in this paper is the one presented by Volodymyr et al. [13]. In this work, authors propose a logger to generate results ready to be processed by process mining techniques with RPA purposes. Considering the requirements listed in Sect. 2 and the related tools mentioned above, a mapping between them was executed resulting in Table 1.

In the classification scheme, for each of the tool or platform, each requirement receives a weight. If the tool provides full support to the requirement, it is weighed as 1. If the tool provides partial support to the requirement (e.g., limitations by payment license), it is weighed as 0.5. If the tool does not provide support to the requirement, it is weighed as 0.

As can be seen in Table 1, all the analyzed tools or platforms provides functionalities to record the keyboard strokes and the name of the application that is being executed on the moment of the capture. The vast majority of them allows the capture of clipboard content. Very close by are those platforms that allow the capture of mouse clicks and screenshots. Slightly above average are the tools or platforms that let the user recording the moment of the capture and send all the information collected to a server. In addition, these tools can be executed in different operative systems. Capturing the mouse position or the

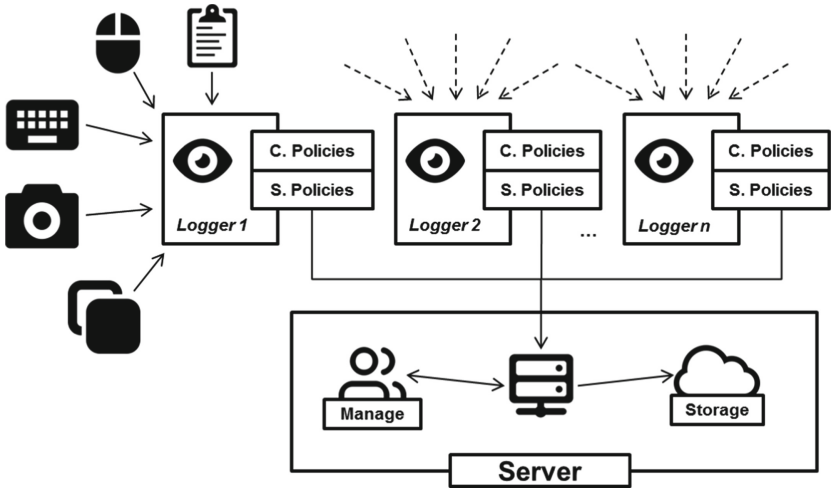

Fig. 1. Logger server

computer's characteristics that is being monitored can be only registered by half of the tools. Less than half of the proposals let the users make remote control of the computer that is being monitored. Finally, only three of the eight tools that have been analyzed have been classified as OpenSource (Fig. 1).

Although some OpenSource projects have been found, results show that most of them are independent modules that can be incorporated into other broader platforms. Thus, they only meet one or more of the defined requirements without completing the full set of them. The two best tools resulting from this classification are Spyrix and SpyAgent. However, they are not OpenSource. In addition, they do not cover important requirements like the possibility of being executed in diverse operating systems or recording the mouse position, among others.

To the best of our knowledge, none of the analyzed tools or platforms satisfies all the requirements defined, this paper presents the foundations of an OpenSource Logger to cover this gap.

4 RPA Logger

4.1 Endpoint Logger

The endpoint logger is focused on gathering enough information for a future RPA analysis. It captures the position and the button type of a mouse click, the keystrokes, and the screen captures. It provides three mechanisms of extension:

- Capture extensions. For specific context, the logger can be extended with a scraper component which adds more information for each event, e.g., web page changes in the context of an RPA project where only web pages are used.
- Capture policies. They are differentiated into two types. First, policies to capture mouse and keyboard events. The current mechanism is to capture one event per mouse click or keystroke. However, it would be interesting to define a policy where a set of keystrokes are grouped in only one event if they are within a defined time window. And secondly, policies to capture screen captures. The current policy capture one image per click or keystroke. However, some scenario that would not afford so many images can decide to make captures in a frequency basis, e.g., one capture every 30 s.
- Send policies. A big amount of data is sent to the central server and, in some context, strict policies must be defined. The current policy sends the event once it occurs. However, in a context where network restrictions apply, a common policy would be to send all the events at the end of the day.

4.2 Central Server

The central server is in charge of storing all the events associated with each monitored computer. In addition, the heavy processing is performed to extract information from the events to be more useful for future RPA analysis.

For example, comparing the similarities between images to detect which ones correspond to the same activity. This comparison may be done by the use of image-similarity techniques [11]. More precisely, an efficient bit-wise comparison [10] between images fingerprints (i.e., short hashes which are obtained from each image in a deterministic way) used to state that two screen captures are related to the same activity according to some prefixed similarity threshold [26]. Another example is to extract patterns or texts from the images applying image processing techniques like Object Character Recognition (OCR) [17].

Data processing in the central server will be performed on demand. At this point, the information being collected does not have to be processed at runtime. Moreover, this way of processing the information can be beneficial to prevent database overloads by avoiding unnecessary iterations.

A simple process, i.e, a teacher that has to consolidate the results of the exams that she marked on her institution website, has been the reference to illustrate how the log should look like. Figure 2 illustrates a simplified log with some of the most interesting fields to be considered. Among them: a global identifier, another one that identifies which computer the capture came from, the timestamp of the capture, the action and the window where the actions are executed.

ID	Computer ID	Timestamp	Action	...	Window
1	1	2020/06/10-0:4:32	Text "Tari Tavarez"	...	Student profile
2	1	2020/06/10-0:4:35	Left click on (100,200)	...	Student profile
3	1	2020/06/10-0:4:50	Text "9.5"	...	Student profile
4	2	2020/06/10-0:5:05	Text "Burton Bertram"	...	Student profile
5	2	2020/06/10-0:5:10	Left click on (115,206)	...	Student profile
6	2	2020/06/10-0:5:20	Left clic on (10, 240)	...	Student
...	...	...	...	...	...

Fig. 2. Log example

5 Conclusions and Future Work

This paper presents the foundations of an OpenSource project which aims to serve as a logger for the analysis phase of RPA projects. After introducing a motivation scenario where the critical requirements have been identified, the closet works related to this proposal have been presented. Although similar proposals have been found, it has been noticed that: (1) they do not cover all the requirements or (2), they are private. Thus, none of the proposals is suitable for giving a solution to the described scenario.

In this context, this paper presents a proposal covering all the aspects mentioned. The proposed solution consists of: (1) a logger capable of collecting information from different events and equipment and sending it to a server and (2), a central server that is responsible for processing this information and converting it into an enriched log so that the data can be processed later.

The immediate future work is focused on preparing the data for processing by data mining techniques. Moreover, an in-depth definition of the requirements will be studied to improve the connection meaning between the requirements and the ones used for the classification scheme. Finally, another important aspect is to manage the data processing itself on the central server.

Acknowledgements. This research has been supported by the Pololas project (TIN2016-76956-C3-2-R) of the Spanish Ministry of Economy and Competitiveness, the Trop@ project (CEI-12-TIC021) of the Junta de Andalucía, and the AIRPA (P011-19/E09) project of the Centro para el Desarrollo Tecnológico Industrial (CDTI) of Spain.

References

1. Aguirre, S., Rodriguez, A.: Automation of a business process using robotic process automation (RPA): a case study. In: Figueroa-García, J.C., López-Santana, E.R., Villa-Ramírez, J.L., Ferro-Escobar, R. (eds.) WEA 2017. CCIS, vol. 742, pp. 65–71. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-66963-2_7
2. Automation Anywhere. Automation Anywhere: Global RPA Solutions (2020). www.automationanywhere.com. Accessed June 2020
3. Bayldon, J.: SharpRPA, a free and open-source RPA solution powered by the.net framework. http://www.taskt.net/. Accessed May 2020
4. Bestxsoftware. Best free keylogger. https://bestxsoftware.com/es/. Accessed May 2020
5. Blue Prism. Blue prism, intelligent RPA platform (2020). www.blueprism.com. Accessed June 2020
6. Dumais, S., Jeffries, R., Russell, D.M., Tang, D., Teevan, J.: Understanding user behavior through log data and analysis. In: Olson, J.S., Kellogg, W.A. (eds.) Ways of Knowing in HCI, pp. 349–372. Springer, New York (2014). https://doi.org/10.1007/978-1-4939-0378-8_14
7. Enríquez, J.G., Jiménez-Ramírez, A., Domínguez-Mayo, F.J., García-García, J.A.: Robotic process automation: a scientific and industrial systematic mapping study. IEEE Access **8**, 39113–39129 (2020)
8. Fung, H.P.: Criteria, use cases and effects of information technology process automation (ITPA). Adv. Robot. Autom. **3**(3), 1–10 (2014)
9. Geyer-Klingeberg, J., Nakladal, J., Baldauf, F., Veit, F.: Process mining and robotic process automation: a perfect match. In: International Conference on Business Process Management, pp. 1–8 (2018)
10. Gusfield, D.: Algorithms on Strings, Trees, and Sequences: Computer Science and Computational Biology. Cambridge University Press, New York (1999)
11. 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: Giorgini, P., Weber, B. (eds.) CAiSE 2019. LNCS, vol. 11483, pp. 446–461. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-21290-2_28
12. Leno, V., Polyvyanyy, A., Dumas, M., La Rosa, M., Maggi, F.M.: Robotic process mining: vision and challenges. Bus. Inf. Syst. Eng. 1–14 (2020)

13. Leno, V., Polyvyanyy, A., La Rosa, M., Dumas, M., Maggi, F.M.: Action logger: enabling process mining for robotic process automation. In: Proceedings of the Dissertation Award, Doctoral Consortium, and Demonstration Track at 17th International Conference on Business Process Management (BPM 2019), Vienna, Austria, pp. 124–128 (2019)
14. OpenRPA. Open source robotic process automation software (2020). https://openrpa.openrpa.dk/. Accessed June 2020
15. Randhawa, A.: Blackcat keylogger. https://github.com/ajayrandhawa/Keylogger. Accessed May 2020
16. Reinkemeyer, L.: Process Mining in Action. Principles, Use Cases and Outlook. Springer, Heidelberg (2020)
17. Singh, S.: Optical character recognition techniques: a survey. J. Emerg. Trends Comput. Inf. Sci. **4**(6), 545–550 (2013)
18. Spyrix Inc.: Spyrix. parental & employees monitoring software. http://www.spyrix.com/. Accessed May 2020
19. Spytech Software and Design, Inc.: Spytech, providing computer monitoring solutions since 1998. https://www.spytech-web.com/spyagent.shtml. Accessed May 2020
20. Taulli, T.: Open source RPA. The Robotic Process Automation Handbook, pp. 259–272. Apress, Berkeley, CA (2020). https://doi.org/10.1007/978-1-4842-5729-6_11
21. UiPath. The UiPath Activities Guide (2020). https://docs.uipath.com/activities. Accessed June 2020
22. UiPath. UiPath enterprise RPA platform, where the future of RPA arrives first (2020). www.uipath.com. Accessed June 2020
23. UiPath. UiPath recording types (2020). http://docs.uipath.com/studio/docs/about-recording-types. Accessed June 2020
24. Aalst, W.: Data science in action. Process Mining, pp. 3–23. Springer, Heidelberg (2016). https://doi.org/10.1007/978-3-662-49851-4_1
25. Willcocks, L., Lacity, M., Craig, A.: Robotic process automation: strategic transformation lever for global business services? J. Inf. Technol. Teach. Cases **7**(1), 17–28 (2017)
26. Wong, C., Bern, M.W., Goldberg, D.: An image signature for any kind of image. In: International Conference on Image Processing, pp. 409–412 (2002)