Action Logger: Enabling Process Mining for Robotic Process Automation

Volodymyr Leno[1,2], Artem Polyvyanyy[1], Marcello La Rosa[1],
Marlon Dumas[2], and Fabrizio Maria Maggi[2]

[1] The University of Melbourne, Parkville, VIC, 3010, Australia
vleno@student.unimelb.edu.au, artem.polyvyanyy@unimelb.edu.au,
marcello.larosa@unimelb.edu.au
[2] University of Tartu, Estonia
marlon.dumas@ut.ee, f.m.maggi@ut.ee

Abstract. This paper presents a tool, called Action Logger, for recording user interface (UI) logs, i.e., logs of user interactions with information systems. By generating output suitable for process mining, the tool aims to introduce process mining methods, techniques, and tools for supporting Robotic Process Automation (RPA) activities, e.g., robot discovery and implementation. Action Logger offers unique capabilities, including logging relevant user actions at a granularity level suitable for RPA, data-awareness, and context-independence.

1 Introduction

Automating repetitive clerical tasks is a common approach to enhance efficiency and reduce defect rates in business processes. Robotic Process Automation (RPA) [4] aims at achieving such automation by creating and running software robots (*bots*) that mimic users interactions with IT systems. Such bots can perform high-volume tasks with higher accuracy and efficiency than a human worker, thus allowing human resources to be directed to other tasks.

The creation of a bot requires an in-depth knowledge of the tasks to be automated, IT systems involved and their UIs, as well as how users interact with them. In current practice, this knowledge is gained via interviews and workshops with the stakeholders, and analysis of unstructured data, e.g., video recordings of users working with the systems [2]. This approach is time-consuming and error-prone, which significantly affects the quality of the bots developed. Therefore, much time is often spent in developing and testing the bots.

Process Mining [1] is a family of techniques, methods, and tools that aim to use information about executed business processes extracted from IT systems' event logs, e.g., ERP and CRM logs, to improve the future processes. Typical information that can be extracted is a flowchart of the process execution, a.k.a. process model, that summarizes the steps of the process as observed in the real world. This flowchart can be annotated with data to build an executable process model, which can potentially be used to generate RPA bots (semi-)automatically.

The adoption of process mining techniques for RPA is hampered by the absence of tools capable of recording UI logs that (i) can be given as input to process mining tools and (ii) contain information at the granularity level suitable for RPA. This paper presents a tool, namely Action Logger, that addresses this gap. The paper discusses the requirements addressed by Action Logger, the tool's architecture, its development status and directions for enhancement. The paper is complemented by a screencast[3], a tutorial[4] and the tool itself.[5]

2 Tool description

This section discusses the requirements that guided the development of UI Logger, its architecture, and a motivating example.

2.1 Functional requirements

To generate UI logs amenable for further RPA-related analysis with process mining, a logger tool should ensure these functional requirements:

- **RQ1: Relevance.** A tool should only record meaningful, value-adding actions. For example, actions of moving the mouse or clicking on the background of a web site should not be recorded as they do not impact the outcome of a task. However, button clicks and URI link clicks are essential actions and, therefore, should be captured.
- **RQ2: Granularity.** A tool should record actions at a level of details sufficient to fully reconstruct the performed task. For example, the logger should be able to differentiate between different types of mouse clicks, e.g., clicking on a button versus clicking a link in a web browser.
- **RQ3: Data-awareness.** In addition to performed actions, a tool should record the data that supports them. These data is crucial in discovering the rule-based decision making logic of the process. The data can also be used to discover data transformations to enhance the quality of the discovered process model. To enable performance analysis, the tool should also record timestamps associated with the performed actions.
- **RQ4: Context-independence.** A tool should record actions in a way that they can be replayed with the same effects on different machines, platforms, under various circumstances and contexts, e.g., different UI layouts.
- **RQ5: Interoperability.** A tool should record UI logs in a format supported by process mining tools, for example CSV and/or XES [3].

To the best of our knowledge, no solution satisfies all the above requirements. Available UI action recording tools, like WinParrot (`www.winparrot.com`) and JitBit Macro Recorder (`www.jitbit.com/macro-recorder`), record low-level actions only, e.g., clickstreams and keystrokes. The recorded actions refer to pixel

[3] The screencast is published at `https://youtu.be/SvPuOdWfByc`.

[4] The tutorial is available at `https://github.com/apromore/RPA_UILogger`.

[5] The tool is available at `https://github.com/apromore/RPA_UILogger/releases`.

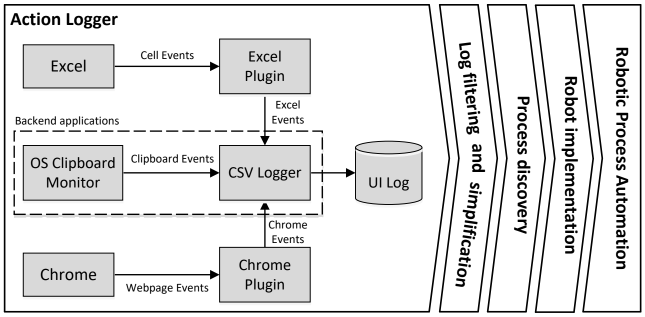


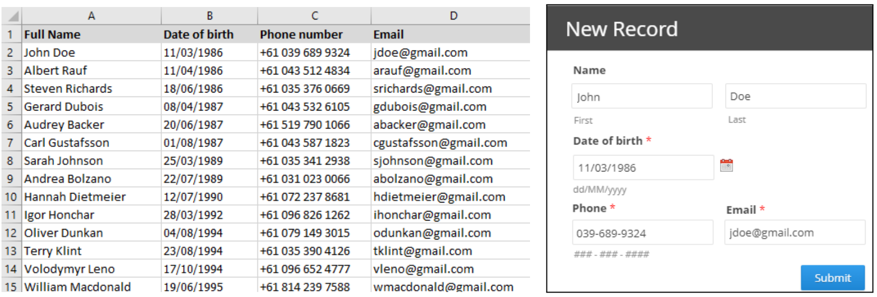
Fig. 1: Action Logger architecture within the RPA pipeline.

coordinates, e.g., click mouse at coordinates 341, 568, that depend on screen resolution and window size. Some tools, like WinParrot, save information regarding the application where the action was performed. However, they do not identify application-specific functionality, e.g., editing or copying a cell in a spreadsheet. Most of the tools do not capture timestamps. However some of them save the delays between actions. None of the existing tools generates files in the format that can be directly consumed by process mining techniques.

RPA tools, like Automation Anywhere (`www.automationanywhere.com`) and UIPath (`www.uipath.com`), provide recording capabilities to generate executable scripts. However, the generated logs are only readable within the environment of the RPA solutions themselves.

2.2 Architecture and features

Action Logger records user actions performed in Excel and Chrome web browser, two often used applications for office tasks. It includes two separate plug-ins, one for each application. The plug-ins are implemented as event listeners and send the information about performed actions as JSON objects to the logging component, which generates and updates the UI log on the fly. To record the actions, the logger uses APIs of the corresponding applications. The browser actions are recorded at the level of the Document Object Model, capturing the involved web elements, e.g., text fields, buttons, and links. The tool also monitors the clipboard to record relevant actions, e.g., copying and pasting of files.



	A	B	C	D
1	**Full Name**	**Date of birth**	**Phone number**	**Email**
2	John Doe	11/03/1986	+61 039 689 9324	jdoe@gmail.com
3	Albert Rauf	11/04/1986	+61 043 512 4834	arauf@gmail.com
4	Steven Richards	18/06/1986	+61 035 376 0669	srichards@gmail.com
5	Gerard Dubois	08/04/1987	+61 043 532 6105	gdubois@gmail.com
6	Audrey Backer	20/06/1987	+61 519 790 1066	abacker@gmail.com
7	Carl Gustafsson	01/08/1987	+61 043 587 1823	cgustafsson@gmail.com
8	Sarah Johnson	25/03/1989	+61 035 341 2938	sjohnson@gmail.com
9	Andrea Bolzano	22/07/1989	+61 031 023 0066	abolzano@gmail.com
10	Hannah Dietmeier	12/07/1990	+61 072 237 8681	hdietmeier@gmail.com
11	Igor Honchar	28/03/1992	+61 096 826 1262	ihonchar@gmail.com
12	Oliver Dunkan	04/08/1994	+61 079 149 3015	odunkan@gmail.com
13	Terry Klint	23/08/1994	+61 035 390 4126	tklint@gmail.com
14	Volodymyr Leno	17/10/1994	+61 096 652 4777	vleno@gmail.com
15	William Macdonald	19/06/1995	+61 814 239 7588	wmacdonald@gmail.com

New Record

Name

John — Doe

First — Last

Date of birth *

11/03/1986

dd/MM/yyyy

Phone *

039-689-9324

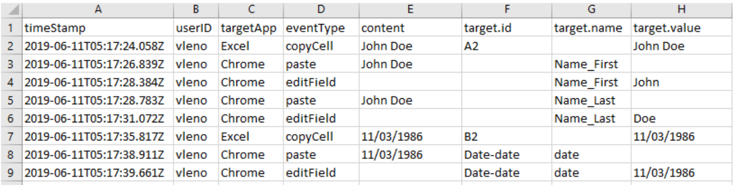
###-###-####

Email *

jdoe@gmail.com

(a) Student records spreadsheet (b) New Record creation form

Fig. 2: Extract of spreadsheet with data that needs to be transferred to a form.



	A	B	C	D	E	F	G	H
1	timeStamp	userID	targetApp	eventType	content	target.id	target.name	target.value
2	2019-06-11T05:17:24.058Z	vleno	Excel	copyCell	John Doe	A2		John Doe
3	2019-06-11T05:17:26.839Z	vleno	Chrome	paste	John Doe		Name_First	
4	2019-06-11T05:17:28.384Z	vleno	Chrome	editField			Name_First	John
5	2019-06-11T05:17:28.783Z	vleno	Chrome	paste	John Doe		Name_Last	
6	2019-06-11T05:17:31.072Z	vleno	Chrome	editField			Name_Last	Doe
7	2019-06-11T05:17:35.817Z	vleno	Excel	copyCell	11/03/1986	B2		11/03/1986
8	2019-06-11T05:17:38.911Z	vleno	Chrome	paste	11/03/1986	Date-date	date	
9	2019-06-11T05:17:39.661Z	vleno	Chrome	editField		Date-date	date	11/03/1986

Fig. 3: Fragment of the recorded UI log after filtering.

The architecture of the tool and the envisaged pipeline for employing the logger for RPA are shown in Fig. 1. The tool stores all the data values used in the context of every recorded action. For example, for an action performed in a spreadsheet, the tool captures the information about the cell, its current value, workbook, and active sheet in which the action took place. The generated logs are stored in the CSV format suitable for process mining investigations.

We also developed a log simplification tool to reduce the size of the UI log in a semantics-preserving manner.[6] The log simplifier works in two steps. First, it converts operating system-level actions into application-level actions (e.g., a get-cell and a copy action are merged into a single copy-cell action). Next, it removes redundant actions, specifically redundant navigation actions in Excel (user selects different cells and then eventually copies the value in a relevant cell), copy actions without corresponding pasting, and copy actions that are later overwritten (douple copying). The log simplifier is implemented as a set of regular expression search-and-replace rules. Some of these expressions are purely control-flow based, e.g., navigation events and double copying, while others are data-aware, e.g., double editing of a text field.

2.3 Example

One typical task automated using RPA is the task of transferring data from one system to another, for example from a spreadsheet to a form of a web-based information system. To demonstrate Action Logger, we use an Excel spreadsheet that contains students' contact details, e.g., full name, date of birth, phone number, and email, and a web form. We manually transferred the data about the first student in the spreadsheet from Excel into the web form, refer to Fig. 2, and recorded all the performed actions using Action Logger. Fig. 3 shows a fragment of the UI log produced by the logger; for the layout considerations, we do not show all the recorded event attributes.

To demonstrate that the recoded UI log is suitable for process mining investigations, Fig. 4 shows the directly follows graph automatically discovered from the log in Fig. 3 using Apromore (`www.apromore.org`)

3 Maturity

To validate Action Logger in practice, we established a cooperation with the University of Melbourne, Australia. In particular, we work closely with the University Services team, which is responsible for the admission and scholarship

[6] The tool is available at `https://github.com/apromore/RPA_SemFilter/releases`.

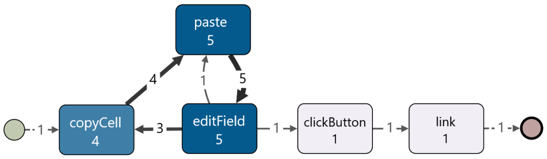


Fig. 4: Process model discovered from the simplified UI log.

allocation processes in the university. The team uses Action Logger to record the workers while performing routine operations and provides feedback with respect to validity (all recorded actions are correct and relevant) and completeness (all the important and relevant actions are recorded) of the recorded logs; note that many of these routines involve work with spreadsheets and web-based frontends of the university IT systems, similar to the example from Section 2.3. We use the collected feedback to improve the tool. We plan to use the recorded logs to support the tasks of robot identification and implementation, in which the University Services team is currently engaged.

4 Conclusions and future work

This paper presented Action Logger, a tool for automatic recording of user interactions with IT systems when performing tasks in a business process. The UI logs recorded by Action Logger are specifically designed to be imported into process mining tools, such as Apromore, thus enabling the analysis of repetitive routines for RPA. In its next releases, we foresee that the tool will support the recording of user interactions with a wider range of software applications (beyond Excel and browsers). We will also explore the use of image recognition techniques and OCR to record user interactions with applications that do not provide suitable APIs for detecting fine-grained events.

Acknowledgments. The authors thank Stanislav Deviatykh for his help in implementing the log simplifier. This research is funded by the Australian Research Council (DP180102839), the Estonian Research Council (IUT20-55), and the European Research Council (project PIX).

References

1. Wil M. P. van der Aalst. *Process Mining - Data Science in Action, Second Edition*. Springer, 2016.
2. B. Agaton and G. Swedberg. Evaluating and developing methods to assess business process suitability for robotic process automation: A design research approach. Master's thesis, Chalmers University of Technology, 2018.
3. C. W. Günther and E. Verbeek. XES standard definition. 2014. `http://www.xes-standard.org/_media/xes/xesstandarddefinition-2.0.pdf`.
4. L. Willcocks and M. Lacity. *Service Automation: Robots and the Future of Work*. Steve Brookes Publishing, 2016.