RESEARCH NOTE

User Behavior Mining

A Research Agenda

Jana-Rebecca Rehse · Luka Abb · Gregor Berg · Carsten Bormann · Timotheus Kampik · Christian Warmuth

Abstract Studying the behavior of users in software systems has become an essential task for software vendors who want to mitigate usability problems and identify automation potentials, or for researchers who want to test behavioral theories. One approach to studying user behavior in a data-driven way is through the analysis of so-called user interaction (UI) logs, which record the low-level activities that a user performs while executing a task. In the paper, the authors refer to the analysis of UI logs as User Behavior Mining (UBM) and position it as a research topic. UBM is conceptualized by means of a four-component framework that elaborates how UBM data can be captured, which technologies can be applied to analyze it, which objectives UBM can accomplish, and how theories can guide the analytical process. The applicability of the framework is demonstrated by three exemplary applications from an ongoing research project with a partner company. Finally, the paper discusses practical challenges to UBM and derives an agenda for potential future research directions.

J.-R. Rehse (✉) · L. Abb
University of Mannheim, Mannheim, Germany
e-mail: rehse@uni-mannheim.de

L. Abb
e-mail: luka.abb@uni-mannheim.de

G. Berg · C. Bormann · T. Kampik · C. Warmuth
SAP Signavio, Berlin, Germany
e-mail: gregor.berg@sap.com

C. Bormann
e-mail: carsten.bormann@sap.com

T. Kampik
e-mail: timotheus.kampik@sap.com

C. Warmuth
e-mail: christian.warmuth@sap.com

1 Introduction

Given the ubiquity of software systems in our digital society and economy, studying the behavior of users in these systems has become an essential task. For instance, software vendors study user behavior to learn how exactly their products are used (Pachidi et al. 2014). By comparing real and intended usage behavior, they can identify problems in functionality (Rubin et al. 2014), such as unintended shortcuts, or usability (Nielsen 1994), such as unused features. Direct insights into user behavior might also inspire new functionalities, such as the automation of routine system tasks by means of robotic process automation (RPA) (van der Aalst et al. 2018; Leno et al. 2021). In addition, researchers study user behavior to better understand, predict, or influence how humans move through a digital environment. For example, they can identify separate user groups based on their behavioral profile (Pachidi et al. 2014), increase employee efficiency through a simplified UI (Astromskis et al. 2015), or support organizations in confronting fraudulent behavior in real-time (Weinmann et al. 2021).

User behavior is typically studied by means of qualitative methods (Amoako-Gyampah 2007), lab experiments (Burton-Jones and Straub 2006), or explicit user feedback (Parks 2012) to collect empirical data. However, these approaches have limitations: they only capture behavior that users actively perceive and cannot study user behavior

over a long period of time (Hoffmann et al. 2019). To address these issues, other analysis approaches rely on quantitative data on the interaction of users with the software, which can be collected automatically. Two popular quantitative approaches for studying user behavior in software systems are web analytics tools such as Google Analytics (Jansen et al. 2023) and click path analysis (Wang et al. 2017), where the actions of users who navigate within a website are collected. However, those techniques are only applicable for browser-based software systems, excluding, e.g., ERP systems. Also, click paths assume a strictly sequential ordering of events which contradicts the highly flexible nature of modern software systems.

An alternative approach to studying user behavior in a data-driven way is through the analysis of so-called user interaction (UI) logs (Dumais et al. 2014), which record the low-level activities that a user performs while executing a task. Each event in a UI log corresponds to a single interaction between the user and the software (Abb and Rehse 2022), such as clicking a button, entering a string into a text field, ticking a checkbox, or selecting an item from a dropdown (Leno et al. 2021). UI logs hence contain high-resolution data on interactions between a user and a graphical user interface (GUI) that can be analyzed to generate insights into user behavior in the software system (Pachidi et al. 2014). The goal of such an analysis is to gain knowledge about and eventually improve the interactions between humans and IT systems. It constitutes a data-driven, non-intrusive approach to studying user behavior and provides a holistic and long-term perspective on software users in real-time (Dumais et al. 2014). In this paper, we refer to the analysis of UI logs as User Behavior Mining (UBM).

In principle, many different techniques can be leveraged for analyzing UI logs (Dumais et al. 2014). However, in this paper, we focus on the application of process mining for realizing UBM, which is a natural point of departure for multiple reasons. User behavior can be conceptualized as a process, i.e., a collections of interrelated activities that collectively lead to a certain outcome (Weske 2019). Whereas the activities in a business process denote single units of work, the "activities" in user behavior relate to the elements of a UI and the actions that are performed on them, such as "click button" (Abb and Rehse 2022). Just like in a business process, these behavioral activities are temporally or logically interrelated, e.g., by succession (button B can only be clicked once checkbox C is enabled) or exclusive choice (once button A was clicked, button B can no longer be clicked). Process mining techniques, such as process discovery, are able to identify these interrelations and visualize them in a chosen modeling language.

In addition to these conceptual analogies between business processes and user behavior, there are also technical analogies. UI logs and process-related event logs have the same structure (Abb and Rehse 2022): a sequence of discrete, atomic events, typically timestamped and specified by additional attributes. Each event can be mapped to an abstract activity. Abstractly speaking, UI logs and event logs both describe "a progression of discrete events that unfolds over time" (Pentland et al. 2020). The fact that UI logs can be specified and stored in the same format as event logs (Abb and Rehse 2022) further facilitates the application of process mining techniques for UBM.

In this paper, we build on extant research in the intersection of business processes and GUIs (Rubin et al. 2014; Leno et al. 2021; van der Aalst et al. 2018) and explore the challenges and opportunities of process mining for UBM. Despite the numerous potentials, data-driven analysis of user behavior through UBM has so far received little attention. The goal of this paper is therefore to position UBM as a research topic and devise an agenda for future UBM research. Therefore, we conceptualize UBM as a four-part framework, which we demonstrate in exemplary applications. After delimiting UBM from related fields, we discuss practical challenges to UBM and derive an agenda for future research opportunities.

2 The UBM Framework

We conceptualize UBM as a four-part framework (Fig. 1). The starting point is UBM data, i.e., UI logs and additional data on the interactions of a user with a software application. This data is processed by UBM technology, i.e., specifically adapted or newly developed data analysis techniques. These technologies aim to realize a UBM objective, which may include analyzing user behavior, developing new user assistance functions, or automating manually executed activities. Where applicable, theories from Information Systems (IS) or Human-Computer Interaction (HCI) can be leveraged to guide the processing of UBM data through UBM technology. In turn, the obtained results may provide empirical evidence on system usage behavior or (by proxy) real-life behavior that helps to develop the theory.

The expected contributions of UBM research are hence fourfold: (1) developing new UBM technology for *processing* UBM data, (2) leveraging existing UBM theories for *guiding* the development process, (3) applying the newly developed technology for *achieving* UBM objectives, and (4) collecting empirical insights into user behavior for *developing* UBM theories.

Fig. 1 UBM Framework
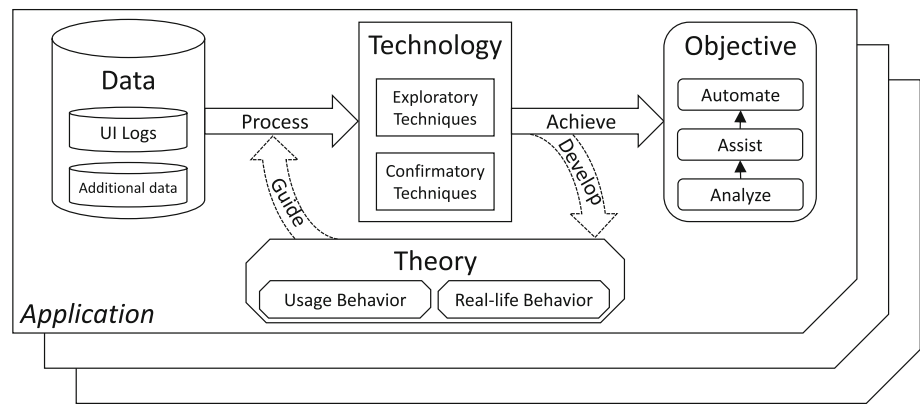
2.1 UBM Data

UBM relies on UI logs, which are high-resolution event logs that record low-level manual activities performed by a user during the execution of a process, task, or procedure in a software system (Abb and Rehse 2022). Just as in a process-related event log, the atomic unit of a UI log is an event, i.e., a single execution of a specific activity that occurred at a certain moment in time (van der Aalst 2016). The activity can be conceptualized in different ways (Abb and Rehse 2022), ranging from single hardware inputs (e.g., *click*) to more abstract descriptions of user actions (e.g., *Login*). The context in which UI events are recorded can be captured by additional attributes. For example, a UI log may contain information about the user who executed the activity, the conceptual task that the activity belongs to, the system that the interaction was recorded on, the application that the user interface belongs to, or the business objects that were modified by the event.

Besides UI logs, UBM may also use additional data to facilitate or enhance the analysis. For example, when analyzing the usability of a newly designed GUI, the results of a user survey could be used to focus the UI log analysis on those regions that the users found to be particularly badly designed. When comparing the behavior of different user groups, user profiles could be categorized to define those groups.

A simple example of a UI log generated from one execution of a task in an ERP system is shown in Table 1. First, the user creates a new sales order, enters values into the four text fields, and saves it. Afterwards, the order's attributes are edited twice more. Each time the "save" button is clicked, a new session / case ID is generated by the system, so that each operation is recorded as a separate case. Activities are represented by the combination of target element and action type.

UI logs exhibit three main characteristics that complicate their analysis: an ambiguous case notion, high variability, and high granularity.

2.1.1 Case notion

As we saw in the example, we have to separate a UI log into groups of interrelated events (so-called cases) in order to analyze user behavior. The problem is that user interactions relate to several context entities simultaneously, such that the UI log may contain several potential case notions:

- *System-generated sessions* allow for analyzing navigation paths between certain points in a system and comparing them with the intended usage paths. However, the sessions cannot connect the log and the business context.
- *Users* allow for tracking an individual user over time and with regard to certain tasks. However, this excludes events that are not associated with a specific user and is too broad for many applications.
- *Business objects*, like orders or invoices, allow for trailing the "lifecycle" of an object independent of the user. However, not all UIs contain these objects and some, e.g., navigation actions, are not associated with a specific business object.
- *Conceptual tasks* allow for tracking activities across multiple objects or applications, which is required in typical automation use cases, such as copying data between applications. However, the concept of a task does not exist on system level and therefore cannot be directly recorded.

2.1.2 Variability

UI logs are characterized by a high variability (Leno et al. 2021; Dev and Liu 2017; Agostinelli et al. 2021). To differentiate between various types of interactions with varying results, the events recorded in a UI log often need to be categorized into event types. These can be low-level distinctions between hardware interactions, such as left and right clicks, or higher-level distinctions, for example

Table 1 UI log of a user creating a sales order, then later changing the amount and price

Case ID	Timestamp	Target element	Action type	Input value
042	30/01 11:31:19	Create Sales Order	click	
042	30/01 11:31:23	Material	input	26
042	30/01 11:31:26	Material	input	\<Tab\>
042	30/01 11:31:31	Amount	input	80
042	30/01 11:31:35	Amount	input	\<Tab\>
042	30/01 11:31:38	Price	input	99
042	30/01 11:31:42	Price	input	\<Tab\>
042	30/01 11:31:52	Customer	input	"DE404"
042	30/01 11:31:58	Save	click	
043	01/02 09:56:50	Sales Order #42	click	
043	01/02 09:56:55	Amount	click	
043	01/02 09:57:02	Amount	input	100
043	01/02 09:57:20	Save	click	
044	03/02 10:20:50	Sales Order #42	click	
044	03/02 10:20:57	Price	click	
044	03/02 10:20:59	Price	input	90
044	03/02 10:21:01	Save	click	

between input events that affect the outcome of a task or process and navigation events that only serve to navigate through the user interface. Since UBM data is captured on the level of interactions with UI elements, each combination of an element and an event type corresponds to a separate activity. Therefore, the number of unique activities in a UI log can easily become very high. Furthermore, software UIs often provide a high degree of flexibility, allowing users to execute activities in an arbitrary order and to revisit previously executed steps. UBM can therefore not rely on clearly defined control-flow relations among the activities, which leads to a high number of unique trace variants.

2.1.3 Granularity

UI logs are also recorded at a high level of granularity, which may be too low for a meaningful analysis. For example, *Ctrl+V* and *right click, paste* correspond to the same user interaction and have the same outcome, but may be recorded as separate activities. On the one hand, the events in UI logs therefore first need to be transformed to an appropriate abstraction level. On the other hand, depending on the use case, many of the recorded events may not be considered relevant for analysis and thus constitute noise. For example, when analyzing usage paths in a particular application, all actions performed on other applications introduce noise. Finally, as a result of the higher granularity, the size of UI logs quickly becomes rather high as well.

2.2 UBM Technology

UBM technology subsumes all data analysis techniques that can be applied to UBM data. Those techniques may be newly developed or adapted from existing data analysis techniques to handle the specific characteristics of UBM data and serve the purposes of UBM. As we have argued above, process mining is a natural point of departure for UBM technology for both conceptual and technical reasons. However, the above-mentioned characteristics of UBM data might complicate the direct application of process mining techniques. In addition, the specific features of UBM, such as the integration of behavioral theories, call for analysis techniques that are not (yet) available in process mining. To further structure UBM technology, we follow the broad distinction of data analysis into exploratory and confirmatory techniques (Janssenswillen and Depaire 2019).

2.2.1 Exploratory Techniques

The goal of exploratory analysis techniques is to uncover new knowledge about the phenomenon described in the data. In a UBM context, this means learning more about the captured user behavior: Which UI elements did they mostly interact with? Which elements are almost never used? What are the main navigation paths through the software? Which paths are the most time-efficient? Are there problems with the usability and if yes, when and where do those problems occur? Can we find differences between the behavior of experienced and inexperienced users? Can we find user behavior that was faulty or even fraudulent?

To answer those questions, we can turn to process mining, which has the goal to investigate previously unknown process behavior to uncover improvement potentials in the process (van der Aalst 2016). Most existing process mining techniques are inherently exploratory (Janssenswillen and Depaire 2019) and in principle suitable to answer the questions. For example, process discovery techniques can show how users navigate through a GUI (Rubin et al. 2014). Process enhancement techniques can discover usability problems like bottlenecks by measuring the duration between events (Dadashnia et al. 2020). Anomaly detection, which utilizes machine learning to find statistical outliers in a dataset (Huo et al. 2021), could identify atypical user behavior in the form of shortcuts, errors, or fraud. Process prediction, which aims to forecast the future behavior of a running process instance (Evermann et al. 2017), can also be applied to UI logs, for example to predict whether a user unexpectedly leaves the application (outcome prediction), what will be their next action (next step prediction), or how long this action will take (execution time prediction).

Process mining techniques may not always be capable to achieve insights of the same quality for a highly variable and fine-granular UI logs as for a regular process event logs. For example, a process model discovered from a UI log could contain hundreds of nodes and thousands of execution variants, making it difficult to read and to understand (Abb et al. 2022). The main challenge of exploratory UBM technology is thus to handle UI logs despite their inherent characteristics.

2.2.2 Confirmatory Techniques

The goal of confirmatory analysis techniques is to validate or falsify an assumed relationship about the phenomenon in the data. Ideally, this assumption helps explain how and why the phenomenon occurs (Recker 2021). In a UBM context, this means finding patterns and causes for the witnessed user behavior: Do users deviate from the intended design? Why do those deviations occur? Why did a user abort their interaction with the software? Will a new shortcut option increase the usage efficiency? Will a new warning feature reduce the amount of fraudulent behavior? Did the user behavior change after relaunching the GUI last year?

To answer those questions, we require data analysis techniques that are able to confirm or reject a hypothesis with statistical significance. These techniques are less common in process mining. One might argue that conformance checking constitutes a confirmatory analysis technique because it postulates certain process behavior (in the form of a process model) and then tests whether the actual process behavior (documented in the event log) deviates from it. However, established techniques, such as alignments (Carmona et al. 2018), only identify and count the number of deviations in the log, without making any general statements about process conformance. Stochastic conformance checking (Leemans et al. 2021), which considers probabilities in the expected process behavior, attempts to address this issue. Applied to a UI log and a probabilistic model of the expected user behavior, it could reveal where and how the users of a software application deviate from the intended design. It could also serve as usage guidance when comparing a user's execution of a task with a process model that captures an expert's execution of the same task.

Other confirmatory process mining techniques that could be adapted to UBM include concept drift detection and causal analysis. The goal of concept drift detection is to detect sudden or gradual changes in the execution of a process. If this is done in a statistically grounded way (Maaradji et al. 2017), it can be applied for example to test whether the user behavior changed significantly after the introduction of a new feature. The goal of causal analysis is to identify causal (instead of merely correlative) relations between, e.g., traces attributes and case outcomes (Bozorgi et al. 2020) or different control flow elements (Leemans and Tax 2022). Applied to a UI log, those techniques could identify discriminating factors between different user behavior patterns. In general, however, the lack of confirmatory process mining techniques demonstrates the need for the development of novel UBM techniques.

2.3 UBM Objective

UBM applications aim to achieve an objective which depends on the software application and the tasks for which it is used. If the user, for example, needs to enter data from a document into a database, the objective should be to automate this task. If, however, the user needs to go through an annual safety training, the objective is to ensure that the training is completed as intended. In the following, we broadly categorize potential UBM objectives into analysis, assistance, and automation.

2.3.1 Analysis

This subsumes all UBM applications that are meant to provide deeper insights into the user behavior. Those insights can be used to identify problems and provide potential solutions. User behavior analysis is closely related to the fields of usability engineering (Nielsen 1994) and data-driven requirements engineering (Hoffmann et al. 2019). Potential objectives in this category include: (1) identifying common usability issues or missing features, (2) pinpointing system areas where user behavior deviates

from system design, or (3) identifying groups of users with similar usage habits and devising strategies to adapt the UI to their preferences.

2.3.2 Assistance

Assistance components are particularly helpful in flexible or complex software applications (Maedche et al. 2016), because they support the user in fulfilling their tasks. In the UBM context, this entails automatically recognizing the user's current actions to enable context-specific features and recommendations. Potential assistance objectives include: (1) simplifying the user's navigation through the system by suggesting the most likely next action or (2) encouraging desired process paths at critical decision points. The latter could occur if, for example, UBM data indicates that certain inputs often lead to process cancellations. If the user enters those values early in the process, they can be warned accordingly, which may reduce processing costs later on.

2.3.3 Automation

In a UI context, automation typically refers to RPA (van der Aalst et al. 2018). In RPA, UBM has two main use cases: (1) it can be used to find repetitive tasks with high automation potential and (2) it is needed to derive automation scripts from records of user activities (Leno et al. 2021).

2.4 UBM Theory

Data, technology, and objective are sufficient to define a UBM application, particularly a technical one. However, if the application concerns the detailed analysis, explanation, or prediction of some aspect of user behavior, we argue that it might benefit from taking a theoretical perspective on user behavior. For this purpose, UBM may leverage theories from IS or HCI, which provide general and causal explanations or predictions about the interactions of humans and IT artifacts (Gregor 2006). Relevant UBM theories may fall into one of two categories:

- Theories on *system usage behavior* concern the behavior of users within the software. For example, the theory of workarounds describes how and why users intentionally deviate from prescribed practices (Alter 2014). Examining workarounds in software usage can help organizations to reduce harmful non-compliance or to identify improvements. So far, workarounds have mainly been studied with qualitative methods, which offer relevant insights into their existence and details, but cannot be scaled to study the frequency of

workarounds within an organization or their evolution over time (Beerepoot et al. 2021). By combining a qualitative research approach with the application of mining techniques, Beerepoot et al. were able to conduct a large-scale and detailed analysis of workaround behavior over time (Beerepoot et al. 2021), which in turn may provide empirical evidence to the theory of workarounds.

- Theories on *real-life behavior* concern the human behavior outside a software. For example, social-cognitive theory (SCT) suggests that behavior changes originate from self-regulatory feedback loops (Bandura 1991). It can be applied to actively promote behavior changes in humans, for example to increase their level of physical activity through an mHealth app (Fallon et al. 2021). Applying UBM for the analysis and comparison of user interactions with such an app would allow tracing the self-regulatory mechanisms that SCT suggests, collecting empirical evidence for SCT and assessing its efficacy to promote behavior change.

UBM theory is part of an inherent feedback loop: Theories can guide the application of UBM technology in determining how to analyze user behavior, for example by scoping the low-level user actions in the log. Simultaneously, UBM technology can provide insights into previously unknown aspects of user behavior, which can be used to develop the theory. We describe both processes in the following.

2.4.1 Guide

Researchers can leverage a theory to explain why users behave in a certain way. For this purpose, they derive one or multiple hypotheses from the theory. A hypothesis formulates a relationship between two or more variables, which is concrete enough to be found as either true or false when tested empirically against the UBM data. Deriving a hypothesis typically requires to specify the theories' constructs in the intended application scope and to operationalize them into a measurable variable (Recker 2021). For example, an extension to the original technology acceptance model postulates that prior usage of an IT system positively influences users' acceptance (Jackson et al. 1997). Applied to predict the acceptance of a new ERP system, we could hypothesize that the "extent to which users use the current information systems (prior usage) will have a positive effect on their behavioral intention to use [the new one]" (Amoako-Gyampah 2007) and operationalize the variable "prior usage" as the average number of weekly interactions between a user and the system in the past year.

Guidance is an outside-in scenario: Applying a theory to the specific UBM context determines the development and application of UBM technology. This may involve both confirmatory and exploratory UBM technology. Confirming (or rejecting) a hypothesis with statistical significance is a more "classical" research approach, but a theory can also be leveraged for exploring the data. For example, the theory of workarounds (Alter 2014) can guide researchers on which components to prioritize when searching for workarounds. Note that the starting point of the guiding process does not have to be a fully developed and widely accepted IS or HCI theory. If no applicable theory is available for a certain UBM context, researchers can also rely on hypotheses that were derived from, e.g., qualitative research methods.

2.4.2 Develop

The insights gained by UBM technology can enrich the theory with empirical findings. Ideally, this can help to further develop the theory. This idea draws on the recent suggestions that digital trace data in general (Berente et al. 2019) and event logs in particular (Grisold et al. 2020) can be used for computationally-driven theory development. Abundantly available trace data about all kinds of studied phenomena give researchers the opportunity to gain a rich understanding of social interactions (Berente et al. 2019). Adapted to user behavior, it allows us to study, for example, the interrelations between different UI features or the determinants for workarounds.

Development is an inside-out scenario: If a pattern is found sufficiently often in the data, it can be considered empirical evidence and may support theory development (Berente et al. 2019) when properly contextualized and complemented by human sense making. Again, this may involve both confirmatory and exploratory UBM technology. If the theory is not yet mature, exploratory techniques can find previously unknown data patterns to support initial theorizing. If the theory is already established, confirmatory techniques can test it in a new context and provide new directions for refinement and enrichment.

3 Exemplary Applications

We demonstrate the applicability of the UBM framework by describing its application in an ongoing research project. Our project partner is a multinational company interested in exploring the potential of UBM for their ERP system product. Combined with adjacent product- and business-oriented initiatives, the broader motivation of this project was to assess to what extent UBM can help to further lift data-driven analysis approaches from the IT system to the human level, thus enabling a socio-technically more precise grasp of business processes and operations. The overall ambition was not to merely demonstrate application potential on a project basis, but rather to assess whether UBM has the potential to play a major role in generally available enterprise software. Under the informed assumption of desirability (market potential), the feasibility-oriented academic research perspectives were augmented by product engineering-oriented assessments. The exemplary applications were designed and assessed from the perspective of generalizability and business scalability in the context of enterprise software products.

The partner company provided us with a real-life UI log that contains data recorded over a period of ten months in ERP systems across multiple administrative departments. The log was automatically generated from their customized system. Its entries reflect a large variety of tasks performed by employees as part of their daily work, including the processing of sales orders and sales documents. Because the system includes multiple processes and UIs, it logs the business object type, the business object identification number, and a session identifier for each event. This triple is used as a case ID. Through the recognition of business object types and identifiers across different UBM logs, the life cycle of, e.g., sales order #42 can be reconstructed across several days, different UIs, and different users, as visualized in Fig. 2.

An excerpt from the UI log that represents the exemplary user actions from Fig. 2 is shown in Table 1. Each event in the log corresponds to a single low-level user action and refers to the target UI element. A case consists of all actions that are executed on a single instance of a business object as part of the same higher-level task, such as filling out a sales order. With this dataset, we realized three exemplary UBM applications that employ different (exploratory) technologies to achieve different objectives.

3.1 Identifying Usage Patterns Through Trace Clustering

As the first step, we wanted to generate a process model that represented the user behavior in the UI log. However, this was impeded by the log's high complexity. The multitude of navigation paths that users can take through the system's interface created an equally high number of process execution variants, which made it virtually impossible to discover a comprehensible process model, even when only looking at traces that relate to one type of task. Trace clustering can address this problem: By grouping together related traces, the log is divided into a number of smaller clusters, each representing one type of user behavior. These clusters introduce an additional level of abstraction between the process and the task level and can ideally be
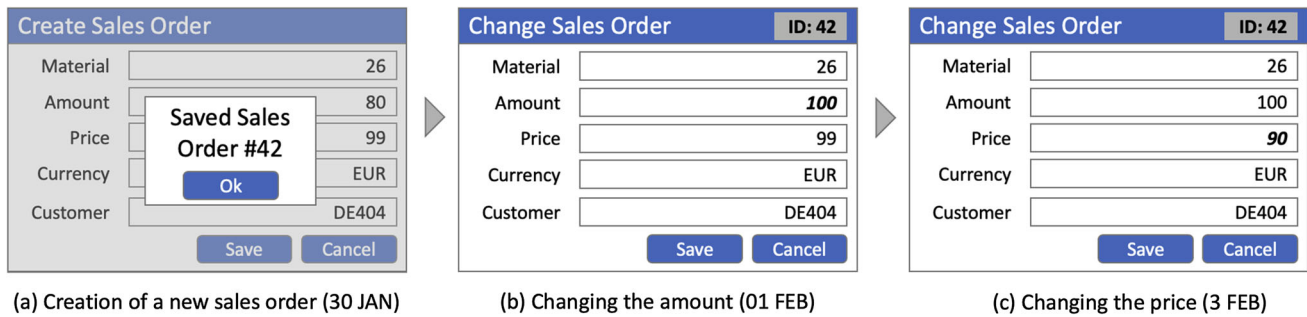
(a) Creation of a new sales order (30 JAN)   (b) Changing the amount (01 FEB)   (c) Changing the price (3 FEB)

Fig. 2 Exemplary visualization of the UI log collection via business object types

visualized in a comprehensible way that is more useful to process analysts.

In our first UBM application, summarized in Fig. 3, we explored the efficacy and applicability of trace clustering by performing a comparative analysis of existing representation learning and trace clustering techniques. We found clusters with multiple execution variants of tasks, like contract cancellation in Fig. 4. For many clusters, we could mine comprehensible process models. For others, the visualizations remained too complex, especially when considering navigation activities.

3.2 Recommending the Next Activity Through Process Prediction

The next objective was the prototypical design of an assistance function that would reduce the navigation complexity. Therefore, we leveraged existing process prediction techniques to predict the activity that a user would most likely execute next, summarized in Fig. 5. The assistance component then provided an according shortcut in the UI. To realize it, we investigated how existing prediction methods performed on our log and designed an according component for the ERP system of the application partner.

Figure 6 shows a prototypical realization of the assistance component. To recommend the most likely next activity, a prediction model is trained on historical UI log data. During a task execution, it is fed with live data and instantaneously returns the label of the most likely next activity. This assistance tool can be extended to also predict the most likely next input values. Currently, the

component recommends the most likely next activity, which is not necessarily the best one for performance or compliance. Next-best activity prediction is technically more challenging because training the model would entail more than recreating the original log. Nevertheless, it would add considerable value to the assistance component, which is why we are currently investigating technical options for its realization.

3.3 Recognizing Non-Compliant Behavior Through Anomaly Detection

The second assistance component was targeted towards solving a concrete business problem. One of the biggest cost factors for our application partner is mitigating non-compliant user behavior, such as maverick buying, which cannot be prevented by the system itself. Therefore, we worked towards an automated real-time recognition of such behavior by means of anomaly detection, summarized in Fig. 7. This application assumes that non-compliant behavior occurs infrequently and can therefore be identified as a statistical anomaly. The goal is to warn the users about the consequences of their actions and achieve a reduction in the overall amount of non-compliant behavior.

By providing real-time feedback about the unusual nature of users' actions, unintended outcomes such as rejections can be avoided, because users can fix the error before it is saved in the system. In the example in Fig. 8, the user mistakenly entered 1000 instead of 100 into the Amount field and now receives a warning that informs them about typical behavior in this case.
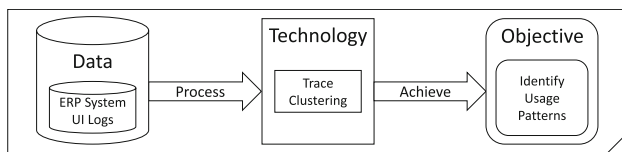
4 Related Work

This section provides an overview of existing methods for analyzing user behavior and mining UI logs, explaining how they relate to UBM.
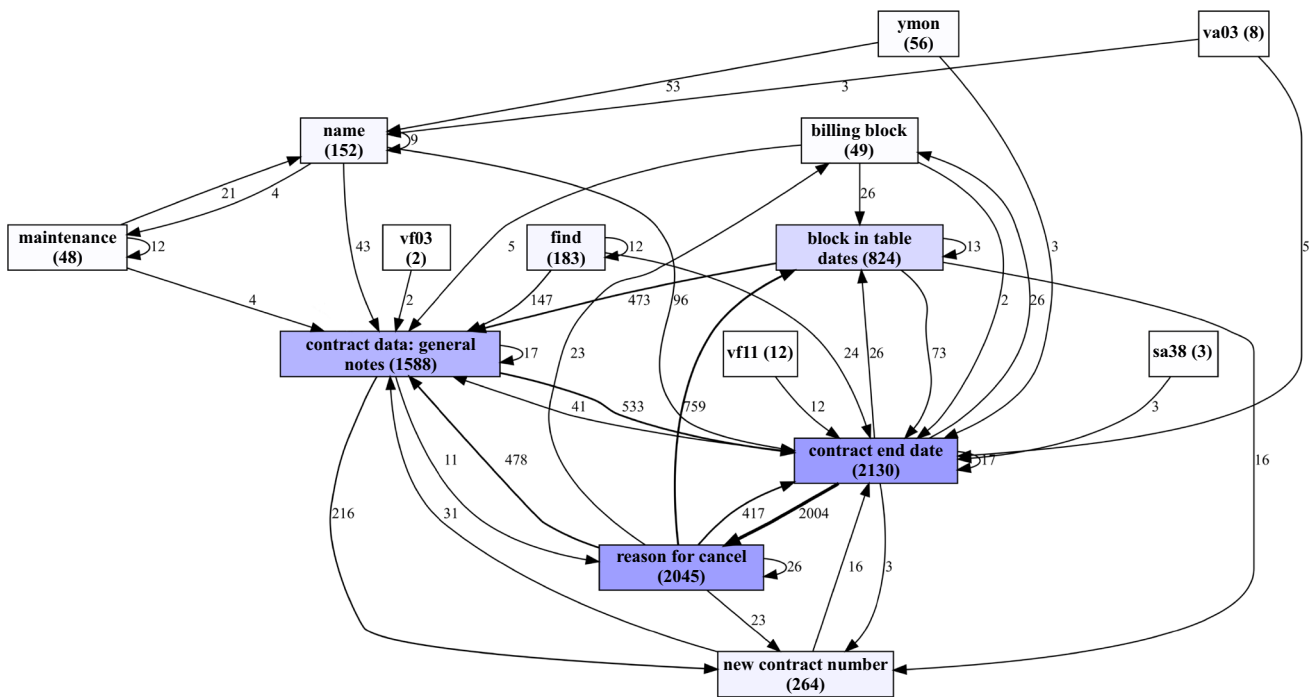

Fig. 3 Applying the UBM framework to identify usage patterns through trace clustering

Fig. 4 Directly-follows-graph for one cluster with traces related to contract cancellation. Nodes are labeled with the IDs of the target UI elements. Even though the cluster contains many trace variants, the main usage paths can still be discerned
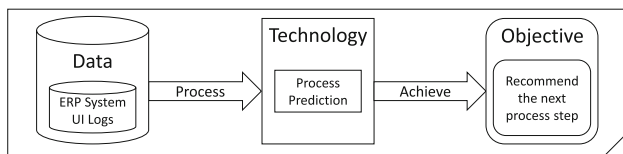

Fig. 5 Applying the UBM framework to recommend the next activity through process prediction

4.1 Analyzing User Behavior in Information Systems

Insights into user interactions with software systems can be gained in multiple ways. Traditionally, user behavior is studied with qualitative empirical methods, such as interviews and questionnaires (Amoako-Gyampah 2007; Lambeck et al. 2014; Stanton et al. 2005; Lim et al. 2015). In industry, many software vendors also gauge the quality of their product by collecting explicit user feedback, e.g., in surveys or feedback forms. However, these methods primarily capture user attitudes and intentions, and they do not allow for the direct analysis of actual system usage. Moreover, they can only reveal usability issues that users actively perceive (Hoffmann et al. 2019; Parks 2012).

Another approach is to conduct experiments, in which users are monitored while they perform specific tasks within a software application in a controlled environment (Burton-Jones and Straub 2006; Parks 2012; Jeong et al. 2020). Although these experiments can provide insights into system usage, they are time-consuming and expensive

and therefore typically limited in scope. Given these shortcomings of traditional methods, data-driven approaches to the analysis of user behavior represent a promising alternative (Dumais et al. 2014).

4.2 Mining User Interaction Logs

Execution data gathered from software systems typically includes the main steps in a process, but does not specify how employees actually perform these steps. Recording and analyzing actions on the UI level can provide a more detailed view into their execution and thus give companies deeper insights into their processes. This lower-level process mining is called *Task Mining* (van der Aalst 2020) or *Desktop Activity Mining* (Maurer et al. 2020). It is related to RPA, which aims to automate manual "white collar" work by scripts that directly replace human data input on UI level (van der Aalst et al. 2018). Mining user data with the aim of automating tasks with RPA is referred to as *Robotic Process Mining* (Leno et al. 2021). Particularly in an industry context, it is sometimes also called Task Mining, blurring the clear distinction of these terms. However, they all refer to the mining of user behavior data with a specific objective in mind. In contrast, we consider UBM a broader research area that covers the mining of UI logs for all possible means. Consequently, it fully entails Task Mining, Robotic Process Mining, and other RPA-related activities.
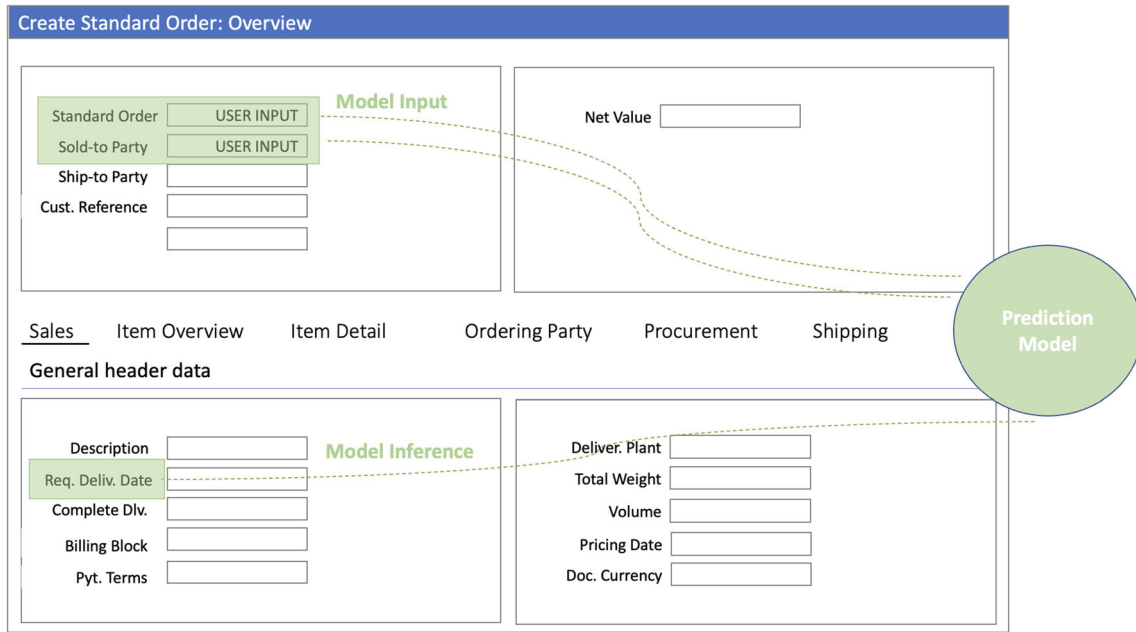
Fig. 6 Process prediction integrated into an ERP User Interface. User interactions streams into a prediction model, which predicts the next UI element the user will most likely interact with
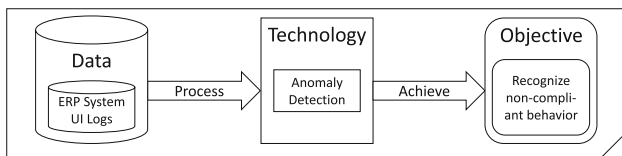

Fig. 7 Applying the UBM framework to find non-compliant behavior through anomaly detection

Another field that aims to mine records of user interactions is *Web Usage Mining*, i.e., the analysis of clickstream and user data recorded during user-website interactions (Srivastava et al. 2000; Ding et al. 2015; Ho et al. 2010; Wang et al. 2017). Unlike task mining, which aims to connect user behavior with a business context, web usage mining's main purpose is to optimize web pages, for example, by adapting their content and structure to users' browsing behavior. Furthermore, interaction logs have

been used as a source of data-driven insights into user behavior in several other domains, such as human-computer interaction (Dumais et al. 2014; Fern et al. 2010), information retrieval (Islamaj Dogan et al. 2009; O'Hare et al. 2016), software usability (Jorritsma et al. 2015; Torok et al. 2015), and visualization (Guo et al. 2016; Dabek and Caban 2017). The logs in these domains can take various forms, but they generally record user interactions at a much lower level of detail than the logs used in a process context.

Finally, mining behavioral models of human or artificial agents has emerged as a line of research within the Multi-Agent Systems (MAS) community (Cao et al. 2012, 2007). In this context, process mining techniques can for example be used to recognize goals for agents and robots (Polyvyanyy et al. 2020). This connection between MAS and Business Process Management (BPM) has been

(a) Creation of a new sales order

(b) Warning dialog informing the user about an anomaly

Fig. 8 Anomaly detection integrated into an ERP User Interface

conceptualized as *Agent System Mining* (ASM) (Tour et al. 2021). ASM is concerned with the mining of agent behavior from event logs, i.e., it focuses on models that represent the perspectives of process participants instead of the organizational view that traditional process mining provides. Although UBM operates on more fine-granular data than ASM, it may be important for ASM because many questions about process participant behavior require the analysis of fine-grained behavioral data that traditional event logs do not provide.

5 Challenges and Future Research

In this section, we discuss some of the research challenges that remain to be addressed to fully leverage the potentials of UBM. The challenges originated in real-world feasibility threats that emerged in an enterprise software productization scenario of UBM. From an implementation and productization perspective, the challenges primarily relate to lifting IT system data analysis approaches to a higher level, at which socio-technical nuances play a more crucial role. This affects both the direct technical feasibility of UBM and the indirect managerial viability of UBM. The technical feasibility of UBM is limited by the higher degree of autonomy that humans have when interacting with a software system and the system's very loose coupling with underlying data models that can guide an analysis. Both factors make it significantly harder to devise minimally viable analysis capabilities that are applicable in a generic context and allow for gaining useful insights without custom-tailored data transformation and analysis tooling. The managerial viability is limited by organizational constraints. For example, UBM has direct, unfiltered access to user behavior, which makes it hard to provide privacy and security guarantees.

From the practical experiences we made with the productization and implementation of UBM, the research agenda was derived by assigning the collected challenges to the respective framework component: data, theory, and technology. Many challenges relate to two components, which is reflected in their assignment. For example, the lack of confirmatory process mining techniques belongs to both theory and technology because new technology needs to be developed to generate new theoretical insights. From the assignment of the challenges, we derived multiple research directions, which could help to address the challenges in the future and together constitute a research agenda for UBM.

Figure 9 summarizes the research challenges as hexagons, each assigned to either a framework component or the intersection of two framework components. Given that our experience mainly stems from a practical application context, most challenges relate to the data and technology component, but we also encountered socio-technical challenges that relate to theory. We do not see any directly objective-related challenges, which we attribute to two reasons: First, the objective typically is the starting point of a UBM application. As such, it determines the configuration of the other three framework components, but not vice versa. Second, the UBM objectives are well established in research and therefore well understood. UBM provides a different methodical approach to achieve them.

In the following, we discuss each challenge and proposed research agenda, organized by the suggested research directions. We do not claim that these challenges are unique to UBM. In fact, many occur also in related fields, for example healthcare process analysis (Munoz-Gama et al. 2022), business process management (Beerepoot et al. 2023), or current process mining research (van der Aalst and Carmona 2022). Nevertheless, those challenges are particularly relevant to UBM because they currently inhibit its practical adoption.

5.1 Conceptualization & Standardization of Logs

5.1.1 Challenges

Currently, UI logs widely differ from one another. This is because there are no standards for specifying and recording them, so that each organization must develop their own tailored solution (Leno et al. 2019). UI logs vary with regard to the number, type, and granularity of recorded events and corresponding attributes (Abb and Rehse 2022). They also typically rely on ad-hoc conceptualizations of elementary notions, such as activities and UI elements. These differences make it difficult to integrate UI logs from different sources (Leno et al. 2021) and challenge the interoperability of data collection and downstream processing tools. Achieving compatibility between tools either requires extensive preprocessing or is entirely infeasible (Leno et al. 2019).

Another compatibility challenge of UI logs is their ambiguous case notion. Although UI logs typically do not lend themselves to a particular case notion, the application of process mining techniques requires to group events by one of the case notion candidates. This inevitable choice causes the loss of context, which makes some insights difficult or impossible to obtain (van der Aalst and Carmona 2022). For example, if we separate the UI log into system-generated sessions, we can no longer analyze the lifecycle of a business object. When events are grouped without considering contextual information, this might also introduce data quality issues like duplicated events (van der Aalst 2019).
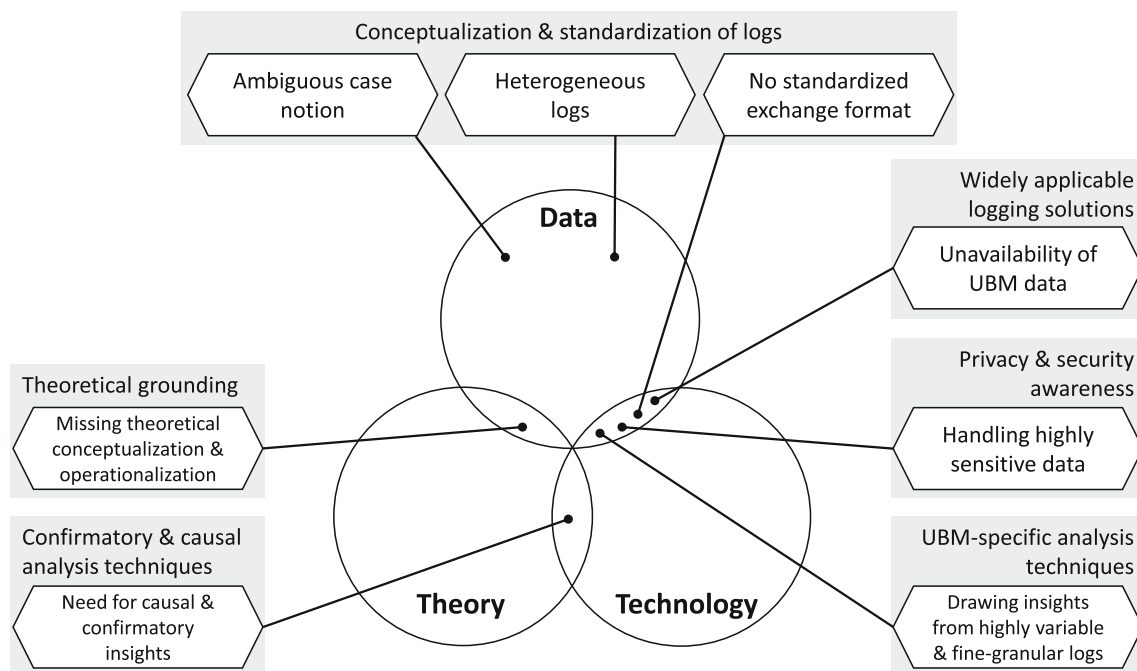
Fig. 9 Challenges & Research Agenda for User Behavior Mining

These compatibility problems are exacerbated by the lack of a standard exchange format for UI logs that is accepted in both academia and industry. We cannot assume that the existing event log standard XES is suitable for this purpose (Wynn et al. 2021), given the differences between UI logs and event logs. In addition, industry solutions rarely use XES even for event logs and instead rely on proprietary solutions or generic formats, such as CSV. The former precludes compatibility, but the latter also poses challenges, including the lack of standardized field names or the inability to handle additional data.

5.1.2 Research Agenda

To address these compatibility issues in UBM data, we suggest a two-step research agenda. First, providing a generic structure for UI logs would increase interoperability and compatibility between different applications. Such a standard for the specification of UI logs would have to be generally applicable in many scenarios and agreed upon by the community. The recently suggested reference model for UI logs (Abb and Rehse 2022) could be a first step in this direction, but its acceptance remains to be seen. Second, building on such a generic structure, a standardized exchange format for UI logs would further simplify the exchange of data between different tools.

One research avenue that would provide both a generic structure and a standardized exchange format for UI logs is the currently discussed OCED meta-model for object-centric process data (Lebherz and Di Ciccio 2022).

Regarding the inherent limitations of the classic case notion (van der Aalst 2019; van der Aalst and Carmona 2022), an object-centric conceptualization is the more appropriate foundation for UI logs because it permits an event to be related to multiple entities/objects simultaneously. For instance, a single user interaction in an object-centric UI log may be related to the user that executes it, the application that it is performed in, and the UI element that is interacted with. The object-centric format also allows for the explicit definition of relations between different types of objects, such as the attribution of UI elements to the software applications that they are a part of. Note that for conventional event logs, models of object relations typically exist in the relational databases of underlying ERP systems, whereas UI element relations may not be formally modeled anywhere, which makes the conceptualization of object-centric UI logs even more challenging. The object-centric event log standard OCED is currently under development (Lebherz and Di Ciccio 2022). Once finalized, it could become the main storage format for UBM data.

5.2 Widely Applicable Logging Solutions

5.2.1 Challenges

There are two general approaches to recording UI logs. The first approach, which combines screen capture and OCR technology with a hardware input recorder, is prevalent in UBM industry tools (Agostinelli et al. 2020). It is

application-agnostic and thus flexible with regard to the number and type of applications. However, it is significantly limited when recording context attributes and has considerable computational overhead. It also produces fractured observations if a task involves multiple users. The second approach uses logging capabilities in the software applications themselves, which allows recording any available internal information (Jimenez-Ramirez et al. 2019). This gets a comprehensive view on user behavior, but limits the scope to one application. This approach requires access to the source code and comes with high implementation effort. Alternatively, application-specific plug-ins can be used. These only have limited access to internal information, but can be used without source code access.

5.2.2 Research Agenda

The development of more widely applicable logging solutions for UI logs would greatly facilitate the acquisition of high-quality UBM data. Currently, logging plug-ins exist for a few widely-used applications, like web browsers and office software (Leno et al. 2019; Agostinelli et al. 2020). Developing plug-ins for other applications and ensuring the compatibility of the different recording solutions would further increase the potential scope of UBM. In addition, developing techniques to leverage data sources other than UI logs, such as screen captures (Martínez-Rojas et al. 2022), would enable a more comprehensive understanding of user activities.

5.3 Privacy and Security Awareness

5.3.1 Challenges

UI logs contains detailed information on human behavior, often in personal settings like the workplace. This makes UBM an ethically sensitive issue: It raises the fundamental question of how tightly an organization should manage its employees. Even more than in BPM and process mining, organizations must determine how they balance trust and control. This involves the handling of sensitive data, both regarding the privacy of the observed individuals and the security of the involved organizations.

When recording UI logs, it is often unavoidable to identify the individual user, especially when logging is coupled with a system's user accounts. User analytics is legally regulated in domains like online shopping, where customers must agree with a site's terms and conditions before entering. However, data privacy becomes an issue when applying UBM in a business context, like an ERP system. This challenge already exists in process-level event logs, especially in sensitive domains like healthcare

(Mannhardt et al. 2019; Munoz-Gama et al. 2022). Due to the highly individualized setting, where a user's behavior is recorded on a per-interaction basis, privacy is also a major challenge for UBM.

UI logs may also contain security-relevant data, such as passwords or other credentials, which must not be logged. Even hashed passwords may compromise system security because the applied hashing method may be easier to crack than the one in the user management system or because access to a UI log may be less restricted than access to this system. In addition, other user-provided data may also be security-relevant, such as communication with customers in security-critical industries. In general, any leaked communications can be exploited for social engineering purposes such as phishing.

5.3.2 Research Agenda

Given these privacy and security risks, UBM is substantially more difficult to scale in practice (e.g., in standard software) than process mining, for which the management of security and privacy concerns is challenging but practically feasible at scale. To address these risks, we suggest a three-part strategy to increase awareness to privacy and security in logging and analysis. First, we should explore security and privacy risks in a multi-perspective way. One perspective concerns legal boundaries and obligations: In which situations is it allowed or forbidden to track user behavior? Which security and privacy measures must be taken to be compliant with, e.g., GDPR? The second perspective is ethical: For which purposes are researchers or organizations willing to conduct UBM? How can they ensure that the tools and data are used responsibly? The third perspective investigates practical questions: Under which circumstances would individuals and organizations give permission for mining user behavior? Which privacy and security measures need to be in place? These perspectives also need to be considered through a cultural lens, as the perception of security and privacy differs considerably between, e.g., North American and European societies (Dourish and Anderson 2006).

Such an exploration can already provide organizational solutions (Mannhardt et al. 2019), such as not capturing user IDs, hashing observed inputs, or implementing strict authorization schemes. Researchers on process mining in healthcare, where privacy of patient data is a similarly relevant problem, have suggested to establish closer collaborations between researchers and organizations to limit the need for exchanging data (Munoz-Gama et al. 2022). Although this is probably helpful for UBM as well, it does not address the practical problem of UBM productization. Therefore, we also need to address this challenge technologically. We advocate for the development or adaptation

of techniques for anonymizing and securely logging UBM data. For the former, it needs to be tested whether approaches for privacy-preserving process mining (Mannhardt et al. 2019) are capable of handling the low granularity and high variability of UI logs. For the latter, which has only marginally been addressed in process mining research (Munoz-Gama et al. 2022), a more comprehensive assessment of security principles for both UI logs and event logs should be conducted. In particular, the application of existing approaches to facilitate system log security (Karande et al. 2017) should be evaluated.

5.4 Drawing Insights from Highly Variable and Fine-Granular Logs

5.4.1 Challenges

Besides the above-discussed ambiguous case notion, UI logs exhibit two main characteristics that limit the applicability of process mining techniques for their analysis: high variability and higher granularity. The high degree of variability in UI logs often precludes the interpretation of process mining results. Our exemplary application demonstrated this problem with a process discovery example. Another example is conformance checking (Carmona et al. 2018),which can be used to analyze the compliance of user behavior with the constraints imposed by normative behavior models. The problem is that existing approaches for behavior modeling and compliance checking assume a well-defined macro-level control flow (Kunze and Weske 2016). Manually creating normative models of highly variable user behavior, e.g., for conformance checking purposes, might therefore not be feasible with the given modeling languages.

The level of granularity, at which UI logs are recorded, presents a related challenge: Fine-granular interactions do not allow for insights into actual user behavior and therefore need to be abstracted. The degree of abstraction depends on the UBM objective: When matching UI logs to process activities, as for example done in task mining, the abstraction level needs to be higher than when analyzing the usage of individual UI features. For fully leveraging UBM, it should be possible to abstract UI logs to the level required by the analysis, which also includes the idenfication of noise.

5.4.2 Research Agenda

To address the variability and granularity of UI logs, we need to develop UBM-specific analysis techniques, inspired by current research in process mining and related areas. Process mining on IoT sensor data is faced with similar challenges (Leotta et al. 2015) and could inspire

new UBM techniques by considering "relaxed" notions of processes and behavior at different levels of abstraction. For example, researchers have already developed new modeling approaches (Seiger et al. 2021) and abstraction techniques (van Zelst et al. 2021) for IoT processes, which could be tested and potentially adapted for UBM. Similarly, variability has been identified as a major challenge in healthcare processes (Munoz-Gama et al. 2022), meaning that the frameworks and analysis techniques could also be helpful for UBM.

Differing levels of event granularity have also been identified as a major problem in process analysis (Beerepoot et al. 2023). It is particularly relevant for loosely specified processes, which closely resemble UIs in their degree of flexibility. There, the authors suggest to work towards flexible granularity levels, such that analysts can select the most appropriate one for their respective use case. Although such an approach might also be applicable to UI logs, there is one important difference to consider: In process analysis, too fine-granular events do not have any process-relevant meaning and therefore need to be abstracted, whereas in UBM, too abstract events loose the connection to the UI and therefore do not convey any insights on user behavior. Hence, a potential solution to this problem might have to be adapted to the UBM context.

Regarding the application of conformance checking, future research should be aimed at creating more flexible and intuitive modeling approaches for user behavior. One option is to rely on declarative modeling approaches (Di Ciccio and Montali 2022) to model constraints or patterns in the expected behavior. These rule-based modeling approaches would alleviate the need to model highly variable behavior, but they would have to be further enhanced to also consider contextual factors (Di Federico and Burattin 2023).

5.5 Confirmatory & Causal Analysis Techniques

5.5.1 Challenges

UBM may contribute to theory development by generating insights on user behavior from UI logs. However, this contribution relies on UBM techniques that can establish or reason about causal relations in the data. This means that it is not sufficient to adapt existing process mining techniques to address the characteristics of UI logs. We also need to develop new UBM techniques that facilitate two novel types of insights. First, we need to be able to test and confirm or reject a hypothesis about the user behavior with statistical significance. An example for such a hypotheses is "This user group behaves differently than the other". Second, we require techniques that establish causal instead of merely correlative relations between different types of

user behavior. For example, we should be able to determine whether the use of feature A eventually triggers the use of feature B.

5.5.2 Research Agenda

To facilitate theoretical insights about user behavior from UI logs, we advocate for the development of novel confirmatory and causal analysis techniques. To confirm or reject hypotheses about user behavior, we require confirmatory process mining techniques. Such techniques can draw on statistical tests that reason about populations, samples, and distributions. A few researchers have suggested to apply such a statistical approach to the analysis of event logs (Janssenswillen and Depaire 2019; Leemans et al. 2021), but much research remains to be conducted. To establish causal relations in the data, we can adopt techniques from causal machine learning, which has also been recently discussed in process mining (Bozorgi et al. 2020; Leemans and Tax 2022). We stress that both hypothesis-testing and causal analysis techniques cannot be limited to user behavior, but must consider contextual factors and their impact on user behavior. It is necessary to utilize contextual knowledge to draw causal conclusions from behavioral data.

5.6 Theoretical Grounding

5.6.1 Challenge

Theories and user behavior differ in terms of their abstraction level. Theories add to the scientific body of knowledge by proposing an explanation of an empirical phenomenon (Recker 2021). Therefore, they must be generic and apply to many situations. By contrast, user behavior is highly contextualized to the software system and its organizational purpose, specific to the situation in which it occurs, and recorded at a high granularity. In this regard, user behavior differs substantially from IT system behavior, which by definition reflects human abstractions of the real-world that were designed with the explicit purpose of information management. Integrating theory and technology for UBM hence requires to match the two components by first *conceptualizing* the user behavior according to the theory in the selected application context and then *operationalizing* the conceptualized behavior such that it becomes measurable in the UI log.

Conceptualization and operationalization are not only required for the behavior itself, but also for its contextual factors. Theories on user behavior typically do not only consider behavior, but also the external factors that influence the behavior. Those factors are important for UBM because they allow for explaining, predicting, and

influencing future user behavior. For example, the theory of workarounds postulates that both situational constraints and individual goals influence the user's perceived need for a workaround (Alter 2014), and hence can be used for preventing future workarounds. UI logs, on the other hand, document the behavior of users in the form of timestamped activities. In addition, they may include the context of those activities, such as information on the user. Therefore, the challenge of theoretical grounding extends to the consideration of contextual factors that might influence user behavior. These factors may occur in the theory as individual constructs, such as the perceived need for a workaround, or as boundary conditions that specify the scope of the theory, such as the level of expertise that a user needs to have to develop a workaround. Because those factors are not directly related to user behavior, we need to find other ways to include them in the data.

This theoretical grounding of human behavior is also required in other IS research contexts. Consider the example of leveraging SCT for promoting more physical activity through an mHealth app (Fallon et al. 2021). To conceptualize mHealth use based on SCT, researchers needed to define the relevant elements of mHealth use (such as monitoring) and relate those to SCT constructs (such as self-regulating behavior) through measures (such as the extent to which users interact with features for self-motivation). As this example shows, existing conceptualizations of theory are dependent on research context and method. This means that they cannot (fully) be reused for UBM applications.

5.6.2 Research Agenda

To provide theoretical grounding for a UBM analysis, we need to conceptualize user behavior and contextual factors by contextualizing the theory to the usage situation. Assuming that only a rich conceptualization of system use (Burton-Jones and Straub 2006) is capable of considering user behavior at the low-level granularity provided by a UI log, this contextualization should involve the system itself, the task to be carried out, and (potentially) the user themselves. Contextual factors contribute to this rich conceptualization because they provide further details on the user or the task.

As a second step, the contextualized constructs (behavior and factors) need to be operationalized by measures. These measures should be computable based on UI logs and take advantage of the detailed information they provide. For example, to measure whether the use of a certain feature triggers the use of another feature, we can inspect directly-follows or eventually-follows relations in the UI log. The operationalization of contextual factors differs from the operationalization of user behavior. Whereas the

latter is already documented in the data and needs to be processed by measures, the former first needs to be documented by variables such that it can be processed by the respective technology. Hence, the operationalization of contextual factors involves enriching the UI log by non-behavioral variables, whose values can be derived from additional UBM data, such as user surveys. Again, the fusion of data and knowledge is required to allow for causal inference.

Conceptualization and operationalization depend on theory, context, and method, meaning that this part of the research agenda needs to be repeated multiple times. However, particularly the conceptualization of a theory in a usage context may offer potentials for reuse. For example, a UI log of an mHealth app could be analyzed using the above-mentioned conceptualization of SCT.

6 Conclusion

In this paper, we introduce User Behavior Mining (UBM) as the (automated) analysis of UI logs to gain knowledge about and eventually improve the interactions between humans and IT systems. We conceptualize UBM by means of a four-part framework, consisting of data, technology, objective, and theory. Using this framework, we present three exemplary UBM applications in an ERP context. We complete the paper by discussing practical challenges that should be addressed in future research to fully leverage the benefits of UBM.

User behavior in software systems is a uniquely positioned topic, because it combines three relevant characteristics: (1) There is a need for its analysis, driven by business or research interests. (2) It is the focus of multiple theories, which can guide this analysis. (3) It is well-documented in UI logs, meaning that technical capabilities can be leveraged for this analysis. In highlighting these characteristics, this paper contributes to further developing UBM in a holistic way. In addition, the UBM framework connects and integrates many ongoing research activities in areas such as task mining or robotic process mining and strives to establish a joint terminology.

Of the four parts of the UBM framework, UBM theory is the least well understood. This is because not many researchers have so far considered the opportunities that process mining methods provide for IS theorizing. Existing research has established process mining as an analytical method to detect and theorize about changing behavior on an organizational level (Grisold et al. 2020), but not on the individual level that is provided by UI logs. Extending this perspective and positioning UBM as a means to analyze the behavior of individual humans offers tremendous opportunities, but it needs to be systematically understood, studied, and applied for these to come to fruition.

References

Abb L, Rehse JR (2022) A reference data model for process-related user interaction logs. Business Process Management. Springer, Heidelberg, pp 57–74

Abb L, Bormann C, van der Aa H, Rehse JR (2022) Trace clustering for user behavior mining. In: European Conference for Information Systems, AIS

Agostinelli S, Lupia M, Marrella A, Mecella M (2020) Automated generation of executable RPA scripts from user interface logs. BPM Forum. Springer, Heidelberg, pp 116–131

Agostinelli S, Marrella A, Mecella M (2021) Exploring the challenge of automated segmentation in robotic process automation. Research Challenges in Information Science. Springer, Heidelberg, pp 38–54

Alter S (2014) Theory of workarounds. Commun Assoc Inf Syst 34(55)

Amoako-Gyampah K (2007) Perceived usefulness, user involvement and behavioral intention: an empirical study of ERP implementation. Comput Human Behav 23(3):1232–1248

Astromskis S, Janes A, Mairegger M (2015) A process mining approach to measure how users interact with software: an industrial case study. In: International Conference on Software and System Process, ACM, p 137-141

Bandura A (1991) Social cognitive theory of self-regulation. Organ Behav Human Decis Process 50(2):248–287

Beerepoot I, Lu X, van de Weerd I, Reijers H (2021) Seeing the signs of workarounds: A mixed-methods approach to the detection of nurses' process deviations. In: Hawaii International Conference on System Sciences

Beerepoot I, Di Ciccio C, Reijers HA, Rinderle-Ma S, Bandara W, Burattin A, Calvanese D, Chen T, Cohen I, Depaire B et al (2023) The biggest business process management problems to solve before we die. Comput Indust 146(103):837

Berente N, Seidel S, Safadi H (2019) Research commentary - data-driven computationally intensive theory development. Inf Syst Res 30(1):50–64

Bozorgi ZD, Teinemaa I, Dumas M, Rosa ML, Polyvyanyy A (2020) Process mining meets causal machine learning: discovering causal rules from event logs. In: International Conference on Process Mining, pp 129–136

Burton-Jones A, Straub D (2006) Reconceptualizing system usage: an approach and empirical test. Inf Syst Res 17(3):228–246

Cao L, Luo C, Zhang C (2007) Agent-mining interaction: an emerging area. Autonomous Intelligent Systems: Multi-Agents and Data Mining. Springer, Heidelberg, pp 60–73

Cao L, Weiss G, Yu P (2012) A brief introduction to agent mining. Auton Agents Multi-Agent Syst 25(3):419–424

Carmona J, van Dongen B, Solti A, Weidlich M (2018) Conformance checking. Springer, Heidelberg

Dabek F, Caban J (2017) A grammar-based approach for modeling user interactions and generating suggestions during the data exploration process. IEEE Transact Visual Comput Graph 23(1):41–50

Dadashnia S, Houy C, Loos P (2020) Usability Mining. Design Science Research Cases. Springer, Heidelberg, pp 155–176

Dev H, Liu Z (2017) Identifying frequent user tasks from application logs. In: International Conference on Intelligent User Interfaces, ACM, p 263-273

Di Ciccio C, Montali M (2022) Declarative process specifications: reasoning, monitoring. Process mining handbook, vol 448. Springer, Heidelberg, pp 495–502

Di Federico G, Burattin A (2023) Do you behave always the same? A process mining approach. In: ICPM workshops, Springer, pp 5–17

Ding AW, Li S, Chatterjee P (2015) Learning user real-time intent for optimal dynamic web page transformation. Inf Syst Res 26(2):339–359

Dourish P, Anderson K (2006) Collective information practice: exploring privacy and security as social and cultural phenomena. Human-Comput Interact 21(3):319–342

Dumais S, Jeffries R, Russell D, Tang D, Teevan J (2014) Understanding user behavior through log data and analysis. Ways of knowing in HCI. Springer, Heidelberg, pp 349–372

Evermann J, Rehse JR, Fettke P (2017) Predicting process behaviour using deep learning. Decis Support Syst 100:129–140

Fallon M, Schmidt K, Aydinguel O, Heinzl A (2021) Feedback messages during goal pursuit: The dynamic impact on mhealth use. In: International Conference on Information Systems, AIS

Fern X, Komireddy C, Grigoreanu V, Burnett M (2010) Mining problem-solving strategies from HCI data. ACM Transact Comput-Human Interact 17(1)

Gregor S (2006) The nature of theory in information systems. MIS Q pp 611–642

Grisold T, Wurm B, Mendling J, Vom Brocke J (2020) Using process mining to support theorizing about change in organizations. In: Hawaii International Conference on System Sciences

Guo H, Gomez S, Ziemkiewicz C, Laidlaw D (2016) A case study using visualization interaction logs and insight metrics to understand how analysts arrive at insights. IEEE Transact Visual Comput Graph 22(1):51–60

Ho S, Bodoff D, Tam K (2010) Timing of adaptive web personalization and its effects on online consumer behavior. Inf Syst Res 22(3):660–679

Hoffmann P, Mateja D, Spohrer K, Heinzl A (2019) Bridging the vendor-user gap in enterprise cloud software development through data-driven requirements engineering. In: International Conference on Information Systems, AIS

Huo S, Völzer H, Reddy P, Agarwal P, Isahagian V, Muthusamy V (2021) Graph autoencoders for business process anomaly detection. Business Process Management. Springer, Heidelberg, pp 417–433

Islamaj Dogan R, Murray G, Névéol A, Lu Z (2009) Understanding pubmed® user search behavior through log analysis. Database

Jackson C, Chow S, Leitch R (1997) Toward an understanding of the behavioral intention to use an information system. Decis Sci 28(2):357–389

Jansen BJ, Jung Sg, Salminen J (2023) Finetuning analytics information systems for a better understanding of users: evidence of personification bias on multiple digital channels. Inf Syst Front pp 1–24

Janssenswillen G, Depaire B (2019) Towards confirmatory process discovery: making assertions about the underlying system. Bus Inf Syst Eng 61(6):713–728

Jeong J, Kim N, Peter H (2020) Detecting usability problems in mobile applications on the basis of dissimilarity in user behavior. Int J Human-Comput Stud 139

Jimenez-Ramirez A, Reijers H, Barba I, Del Valle C (2019) A method to improve the early stages of the robotic process automation lifecycle. Advanced Information Systems Engineering. Springer, Heidelberg, pp 446–461

Jorritsma W, Cnossen F, Dierckx R, Oudkerk M, Van Ooijen P (2015) Pattern mining of user interaction logs for a post-deployment usability evaluation of a radiology PACS client. Int J Med Inform 85

Karande V, Bauman E, Lin Z, Khan L (2017) Sgx-log: Securing system logs with sgx. In: Computer and communications security, ACM, p 19-30

Kunze M, Weske M (2016) Behavioural models - From modelling finite automata to analysing business processes. Springer, Heidelberg

Lambeck C, Müller R, Fohrholz C, Leyh C (2014) (Re-)Evaluating user interface aspects in ERP systems – an empirical user study. In: Hawaii International Conference on System Sciences, pp 396–405

Lebherz J, Di Ciccio C (2022) OCED meta-model presentation, XES/OCED symposium, ICPM 2022. https://icpmconference.org/2022/wp-content/uploads/sites/7/2022/12/OCED_Symposium_intro.pdf, accessed 25 Oct 2022

Leemans S, Tax N (2022) Causal reasoning over control-flow decisions in process models. Advanced Information Systems Engineering. Springer, Heidelberg, pp 183–200

Leemans S, van der Aalst W, Brockhoff T, Polyvyanyy A (2021) Stochastic process mining: earth movers' stochastic conformance. Inf Syst 102(101):724

Leno V, Polyvyanyy A, La Rosa M, Dumas M, Maggi F (2019) Action logger: enabling process mining for robotic process automation. In: BPM Demos

Leno V, Polyvyanyy A, Dumas M, La Rosa M, Maggi FM (2021) Robotic process mining: vision and challenges. Bus Inf Syst Eng 63(3):301–314

Leotta F, Mecella M, Mendling J (2015) Applying process mining to smart spaces: perspectives and research challenges. CAiSE Workshops. Springer, Heidelberg, pp 298–304

Lim S, Bentley P, Kanakam N, Ishikawa F, Honiden S (2015) Investigating country differences in mobile app user behavior and challenges for software engineering. IEEE Transact Softw Eng 41(1):40–64

Maaradji A, Dumas M, La Rosa M, Ostovar A (2017) Detecting sudden and gradual drifts in business processes from execution traces. IEEE Transact Knowl Data Eng 29(10):2140–2154

Maedche A, Morana S, Schacht S, Werth D, Krumeich J (2016) Advanced user assistance systems. Bus Inf Syst Eng 58(5):367–370

Mannhardt F, Koschmider A, Baracaldo N, Weidlich M, Michael J (2019) Privacy-preserving process mining. Bus Inf Syst Eng 61(5):595–614

Martínez-Rojas A, Jiménez-Ramírez A, Enríquez JG, Reijers HA (2022) Analyzing variable human actions for robotic process automation. In: Di Ciccio C, Dijkman R, del Río Ortega A, Rinderle-Ma S (eds) Business process management. Springer, Cham

Maurer B, Al-Taie C, Zimmermann P, Linn C, Werth D (2020) Aufgabenfelder und Einsatzmöglichkeiten von Desktop Activity Mining. HMD Prax Wirtschaftsinform 57(6):1163–1172

Munoz-Gama J, Martin N, Fernandez-Llatas C, Johnson OA, Sepúlveda M, Helm E, Galvez-Yanjari V, Rojas E, Martinez-Millana A, Aloini D et al (2022) Process mining for healthcare: characteristics and challenges. J Biomed Inform 127(103):994

Nielsen J (1994) Usability engineering. Morgan Kaufmann, Burlington

O'Hare N, Juan P, Schifanella R, He Y, Yin D, Chang Y (2016) Leveraging user interaction signals for web image search. In: International Conference on Research and Development in Information Retrieval, ACM, pp 559–568

Pachidi S, Spruit M, van de Weerd I (2014) Understanding users' behavior with software operation data mining. Comput Human Behav 30:583–594

Parks N (2012) Testing & quantifying ERP usability. In: Annual Conference on Research in Information Technology, ACM, p 31-36

Pentland BT, Recker J, Wolf JR, Wyner G (2020) Bringing context inside process research with digital trace data. J Assoc Inf Syst 21(5):5

Polyvyanyy A, Su Z, Lipovetzky N, Sardiña S (2020) Goal recognition using off-the-shelf process mining techniques. In: International Conference on Autonomous Agents and Multiagent Systems, ACM, pp 1072–1080

Recker J (2021) Scientific research in information systems, 2nd edn. Springer, Heidelberg

Rubin V, Mitsyuk A, Lomazova I, van der Aalst W (2014) Process mining can be applied to software too! In: International Symposium on Empirical Software Engineering and Measurement, ACM

Seiger R, Kühn R, Korzetz M, Aßmann U (2021) Holoflows: modelling of processes for the internet of things in mixed reality. Softw Syst Model 20(5):1465–1489

Srivastava J, Cooley R, Deshpande M, Tan PN (2000) Web usage mining: discovery and applications of usage patterns from web data. SIGKDD Explor 1:12–23

Stanton J, Stam K, Mastrangelo P, Jolton J (2005) Analysis of end user security behaviors. Comput Secur 24(2):124–133

Torok L, Pelegrino M, Lessa J, Trevisan D, Vasconcelos C, Clua E, Montenegro A (2015) Evaluating and customizing user interaction in an adaptive game controller. In: International Conference of Design, User Experience, and Usability, Springer, Heidelberg

Tour A, Polyvyanyy A, Kalenkova A (2021) Agent system mining: vision, benefits, and challenges. IEEE Access 9

van der Aalst W (2016) Process mining: data science in action, 2nd edn. Springer, Heidelberg

van der Aalst W (2019) Object-centric process mining: dealing with divergence and convergence in event data. Software Engineering and Formal Methods. Springer, Heidelberg, pp 3–25

van der Aalst W (2020) On the pareto principle in process mining, task mining, and robotic process automation. International Conference on Data Science. SciTePress, Technology and Applications, pp 5–12

van der Aalst W, Carmona J (2022) Scaling process mining to turn insights into actions. Process mining handbook. Springer, Heidelberg, pp 495–502

van der Aalst W, Bichler M, Heinzl A (2018) Robotic process automation. Bus Inf Syst Eng 60(4):269–272

Wang G, Zhang X, Tang S, Wilson C, Zheng H, Zhao BY (2017) Clickstream user behavior models. ACM Transact Web 11(4):21:1–21:37

Weinmann M, Valacich J, Schneider C, Jenkins JL, Hibbeln MT (2021) The path of the righteous: using trace data to understand fraud decisions in real time. MIS Q

Weske M (2019) Business process management: concepts, languages, architectures, 3rd edn. Springer, Heidelberg

Wynn MT, Lebherz J, van der Aalst W, Accorsi R, Di Ciccio C, Jayarathna L, Verbeek H (2021) Rethinking the input for process mining: Insights from the XES survey and workshop. In: ICPM workshops, Springer, Heidelberg

van Zelst S, Mannhardt F, de Leoni M, Koschmider A (2021) Event abstraction in process mining: literature review and taxonomy. Granul Comput 6(3):719–736