A Method for Predicting Workarounds in Business Processes

Short Paper

Sven Weinzierl
FAU Erlangen-Nürnberg,
Fürther Str. 248, 90429 Nürnberg,
Germany
sven.weinzierl@fau.de

Christian Bartelheimer
Paderborn University,
Warburger Str. 100, 33098 Paderborn,
Germany
christian.bartelheimer@upb.de

Sandra Zilker
FAU Erlangen-Nürnberg,
Fürther Str. 248, 90429 Nürnberg,
Germany
sandra.zilker@fau.de

Daniel Beverungen
Paderborn University,
Warburger Str. 100, 33098 Paderborn,
Germany
daniel.beverungen@upb.de

Martin Matzner
FAU Erlangen-Nürnberg,
Fürther Str. 248, 90429 Nürnberg,
Germany
martin.matzner@fau.de

Abstract

Workarounds are performed intentionally by employees to bypass obstacles constraining their day-to-day work. These obstacles manifest from latent misfits in the interplay of information systems, organizational structure, and human agency. While workarounds are often mandatory for employees to perform their work, they can yield positive and negative effects on an organization’s performance. Process managers are supposed to identify workarounds early, promoting their positive while reducing their negative consequences. While related research has touched upon detecting workarounds in event logs that include data on completed processes, little is known on how to predict workarounds in a running business process. We set out to design a workaround prediction method using a deep learning approach. The IT artifact enables process managers to proactively intervene if workarounds are about to emerge in a business process, reducing their adverse effects while supporting organizational learning and process innovation.

Keywords: workarounds, business processes management, process mining, predictive business process monitoring, deep learning, machine learning

Introduction

Business process management (BPM) provides methods and tools for organizations to efficiently and effectively manage and continuously improve their business processes (van der Aalst 2013). While business processes are viewed as standard operating procedures that represent best practices of day-to-day work in
A Method for Workaround Prediction

Emergence of Workarounds in Business Processes

A business process is a “collection of inter-related events, activities, and decision points that involve a number of actors and objects, and that collectively lead to an outcome that is of value to at least one customer” (Dumas et al. 2018, p. 6). This way, organizations often aspire to monitor and control activities of employ-
employees (Ignatiadis and Nandhakumar 2009). Nevertheless, employees implement workarounds regularly as “conscious adaptations of work activities that are not expected or specified to be changed in this manner” (Laumer et al. 2017, p. 335). When employees overcome, bypass, or minimize the impact of obstacles or structural constraints (Alter 2015) in day-to-day work to achieve organizational or individual goals (Ejenejfjäll and Ågerfalk 2019), inefficiencies (Boudreau and Robey 2005), hazards, and process drift can occur over time (Pentland et al. 2020). For example, employees who cannot use a specific software tool to store information needed in later activities by other employees might respond to this obstacle by simply skipping the step and thus not providing the information or by adding an activity to create an additional file that is then sent around via e-mail. Recent IS research conceptualizes seven workaround types that can be performed by employees: violated responsibility, manipulated data, repeated activity, substituted activity, interchanged activity, bypassed activity, and added activity (Weinzierl et al. 2022).

Figure 1 depicts the process of workaround institutionalization in organizations that can take months to years (Alter 2014). The performance and institutionalization of workarounds can affect other employees directly or indirectly, depending on their type of collaboration in an organization (Laumer et al. 2017). The interconnectedness of employees in an organization and the inherent dynamism of workarounds can create ripple-effects that transform organizations as socio-technical systems (Wolf and Beverungen 2019). Therefore, it is essential to develop methods for predicting workarounds in business processes that enable proactive workaround management, avoiding their uncontrolled diffusion in organizations.

Process Mining for Predicting Workarounds

As a sub-discipline of BPM, process mining (PM) comprises a comprehensive set of data-driven methods “to discover, monitor, and improve […] processes […] by extracting knowledge from event logs readily available in today’s (information) systems” (van der Aalst et al. 2011, p.172). PM allows organizations to understand better how work is performed and evinces (hidden) change for managers (Mendling 2020). Concerning workaround analysis in business processes, few papers have introduced methods for detecting workarounds from event log data. For example, Outmazgin and Soffer (2016) proposed a rule-based compliance checking method, Beerepoot et al. (2021) applied a multi-perspective conformance-checking method, and Weinzierl et al. (2022) presented a method relying on deep learning (DL). Even though these methods propose first approaches to detect specific patterns in historical event log data1 that refer to specific workaround types, none of them affords to predict them in running business processes.

Beyond pattern detection, a different PM research stream, namely predictive business process monitoring (PBPM) (Maggi et al. 2014), provides, among others, methods for predicting future (process) outcomes in running business processes. Outcome predictions refer to the classification of running business process instances linked to a given set of possible, future categorical outcomes (Teinemaa et al. 2019). These outcomes are often binary and reflect the violation of normal behavior. Typical questions answered with outcome-oriented PBPM methods refer to identifying customer complaints or canceled orders. Based on outcome-

---

1Historical event log data represents a sample of past event log data taken from a process-based event stream.
oriented predictions, the performance of business processes can be improved, for example, to proactively reduce risk or exploit an opportunity for improvement (Márquez-Chamorro et al. 2017). As the outcomes of PBPM methods are typically specified with rules (Maggi et al. 2014), expressing process violations (Teinemaa et al. 2019), existing PBPM methods are conceptually unable to predict workarounds in running process instances.

**Research Method**

The DSR paradigm enables IS researchers to pursue a dual mission by developing innovative IT artifacts (March and Smith 1995) that enhance the IS knowledge base while solving relevant business problems (Baskerville et al. 2018; Sein et al. 2011). According to Hevner et al. (2004), IT artifacts are built in iterative cycles of design and evaluation. In this paper, we instantiate the DSR methodology proposed by Peffers et al. (2007) with a problem-centered initiation (see Figure 2), to develop a method for predicting workarounds as a new class of IT artifacts (March and Smith 1995). We position the resulting IT artifact as an exaptation (Gregor and Hevner 2013), adapting knowledge on related outcome-oriented PBPM methods (e.g., Teinemaa et al. 2019).

![Figure 2. Research Process to Design and Evaluate the IT Artifact (Adapted From Peffers et al. 2007).](image)

**Artifact Description**

Our objective was to design a DL-based method to predict types of workarounds in business processes, enabling effective and efficient workaround management to implement innovation and prevent negative consequences. As proposed in previous research on PBPM (Teinemaa et al. 2016), the method is structured into an offline and an online component. Each component comprises two phases, as shown in Figure 3. The offline component receives an event log as input. The event log includes process and label data. In the first phase, the method transforms the event log into a data set (including process data) and a set of workaround labels. One workaround label refers to one workaround type (Weinzierl et al. 2022). Subsequently, for each workaround type, the method learns a deep neural network (DNN) model based on the input data (second phase). In the third phase, the online component receives a running process instance as input and transforms it into a data set instance. Finally (phase four), the method applies the learned DNN models to the data set instance and predicts if one or more workaround types will occur in the running process instance.

**Offline Component**

In the first phase of the offline component, the method receives an event log \( L \) as an input and transforms it into two sets: (1) a data set \( X \) and (2) a set of workaround labels \( Y = \{Y_1, Y_2, \ldots, Y_n\} \), where \( n \) is the number of workaround labels. This transformation process comprises the following three steps. First, an event log \( L \) is loaded. \( L \) is a set of traces, and in turn, a trace \( \sigma \) is a sequence of events representing a process instance. An event \( e \) of the trace \( \sigma \) is a tuple \((r, a, t, d_1, d_2, \ldots, d_n, l_1, l_2, \ldots, l_n)\), where \( r \) is the process instance id, \( a \) is the activity, \( t \) is the timestamp, and \( d_n \) and \( l_n \) denote for example the \( n^\text{th} \) data attribute and workaround label assigned.
to the event \( e \). Second, the data set \( X \in \mathcal{R}^{s \times q \times v} \) is created based on the event log \( L \), where \( s \) is the number of prefixes, \( q \) is the number of time steps of the longest prefix in \( L \), and \( v \) is the number of attributes. A prefix is a part of a trace or the entire trace itself. For example, for a given trace \( e_1 = (e_1, e_2, e_3) \), the respective prefixes are \( (e_1) \), \( (e_1, e_2) \), and \( (e_1, e_2, e_3) \). The method creates prefixes of traces because it predicts the appearance of workarounds at each time step in a running process instance. The number of attributes \( v \) is the size of vector \( x \) that represents \( e \). The vector \( x \) includes information on the event’s activity attribute and its additional data attributes. Each attribute is encoded depending on its type. For example, the activity attribute is categorical and one-hot encoded because DNNs typically require numerical data for calculating forward and backward propagation. In contrast, a numerical data attribute is scaled in the range \([0, 1]\) to ensure an efficient convergence during model learning. Third, the method creates the set of workaround labels \( Y \) based on the event log \( L \). For each workaround label, referring to a workaround type, the method creates a workaround label \( Y \in \mathcal{R}^{s \times u} \), where \( s \) is the number of prefixes, and \( u \) denotes the binary output for prefixes, that is, will one of the described seven workaround types appear in the prefix or not.\(^2\)

In the second phase of the offline component, for each workaround type, the method learns the internal parameters of a DNN model \( M_i \) based on the data set \( X \) and the corresponding workaround label \( Y_i \), where \( i \) is the workaround type of interest. The architecture of each DNN model \( M_i \) consists of three layers: input, hidden, and output. The input layer receives each prefix of the data set \( X \) and transfers it to the long short-term memory (LSTM) cells (Hochreiter and Schmidhuber 1997) \( LSTM^+ \) (forward direction) and \( LSTM^- \) (backward direction) of the bi-directional long short-term memory (BiLSTM) (hidden) layer (Graves et al. 2013). DNN models with a BiLSTM layer have shown promising prediction results for outcome-oriented PBPM (e.g., Wang et al. 2019; Weinzierl et al. 2021). This can be attributed to the fact that these layers use two LSTM cells modeling the temporal aspect of (sequential) process traces explicitly from both directions. The output of both cells—that is, hidden state vectors \( h_+ \) and \( h_- \)—are concatenated, and a softmax activation is applied on it in the output layer. As a result, the model returns for each prefix a probability distribution vector \( o \). The internal parameters of a DNN model \( M_i \) are updated batch-wise and over 100 epochs (i.e., learning iterations). For each epoch, the data set \( X \) is partitioned into batches of 32 prefixes. While a loss function calculates for each prefix of a batch the cross-entropy between its probability distribution vector \( o \) and assigned ground truth label \( y \), a cost function sums up the output of the loss functions over the prefixes of this batch. Given the output of the cost function, the gradient descent algorithm NADAM updates the model’s internal parameters. After completing the last epoch, the model \( M_i \)’s internal parameters are adjusted.

**Online Component**

In the third phase, the method’s online component receives a running process instance \( \sigma_r \), as input and transforms it into a data set instance \( X \in \mathcal{R}^{t \times q \times v} \), representing a prefix of a trace, where \( q \) and \( v \) are defined as in the offline component. In contrast to the method’s event log transformation (see phase one), workaround labels are not created as the method predicts the appearance of workaround types based on prefixes in the

\(^2\)We refer to the conceptualization of workaround types by Weinzierl et al. (2022).
in IS research, and a definition can be found in e.g., Breuker et al. (2016).

Workaround types. As a running process instance $i$, a probability distribution vector $o_i$ is returned. From this vector, a workaround prediction $o_i$ is retrieved through the function $\arg\max()$. Consequently, the method provides a prediction for each workaround type $i$, indicating whether the type $i$ will appear in the running process instance $\sigma_r$.

**Preliminary Results**

To demonstrate the applicability of our method, we use a publicly available event log$^3$. The underlying use case is a ticket management process of a help desk in an Italian software company. A trace in the event log represents each case handled by the help desk employees. We removed process instances, which only appear once, from the original event log to exclude arbitrary undesired deviating phenomena that are conceptually distinct from workarounds. The resulting event log includes 20,322 events in 4,445 process instances comprising 91 different process variants. The event log contains ten different activities. Once an issue is resolved, the ticket is closed, and the process instance is terminated. After obtaining a general understanding of the context, we manually identified and labeled the types of workarounds occurring in the event log based on a consensus decision and added the labels to each corresponding event in the event log.$^4$

Furthermore, we split the event log into a 67% training and 33% test set. Additionally, the last 10% of each training set was used as a validation set. Based on the validation set, we applied early stopping after 10 epochs to avoid overfitting of the DNN models. For each BiLSTM layer of the DNN models, we set the internal size, the activation, and dropout rate to 100, $\tanh$, and 0.2, respectively. Moreover, we calculated machine learning metrics to assess the models’ predictive performance. We calculated the $\text{Support}$, $\text{Precision}$, $\text{Recall}$, and $\text{F1-Score}$ per class. Across both classes, we calculated the $AUC_{\text{ROC}}$ per model. These measures are common in IS research, and a definition can be found in e.g., Breuker et al. (2016). As a baseline, we used a random forest classifier with default hyperparameter setting. Table 1 shows the predictive performance per model, representing our preliminary results.

<table>
<thead>
<tr>
<th>Model</th>
<th>Class</th>
<th>Support</th>
<th>Precision</th>
<th>Recall</th>
<th>F1-Score</th>
<th>$AUC_{\text{ROC}}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$M_{\text{added}}$</td>
<td>Workaround</td>
<td>23</td>
<td>0.881 (0.750)</td>
<td>0.391 (0.261)</td>
<td>0.530 (0.387)</td>
<td>0.960 (0.846)</td>
</tr>
<tr>
<td></td>
<td>No Workaround</td>
<td>5,196</td>
<td>0.997 (0.997)</td>
<td>1.000 (1.000)</td>
<td>0.998 (0.998)</td>
<td></td>
</tr>
<tr>
<td>$M_{\text{bypassed}}$</td>
<td>Workaround</td>
<td>284</td>
<td>0.983 (0.947)</td>
<td>0.602 (0.570)</td>
<td>0.747 (0.712)</td>
<td>0.966 (0.947)</td>
</tr>
<tr>
<td></td>
<td>No Workaround</td>
<td>4,935</td>
<td>0.978 (0.976)</td>
<td>1.000 (0.998)</td>
<td>0.988 (0.987)</td>
<td></td>
</tr>
<tr>
<td>$M_{\text{interchanged}}$</td>
<td>Workaround</td>
<td>40</td>
<td>0.870 (0.333)</td>
<td>0.500 (0.025)</td>
<td>0.635 (0.047)</td>
<td>0.883 (0.877)</td>
</tr>
<tr>
<td></td>
<td>No Workaround</td>
<td>5,179</td>
<td>0.996 (0.992)</td>
<td>1.000 (0.999)</td>
<td>0.998 (0.996)</td>
<td></td>
</tr>
<tr>
<td>$M_{\text{repeated}}$</td>
<td>Workaround</td>
<td>1,722</td>
<td>0.878 (0.843)</td>
<td>0.635 (0.640)</td>
<td>0.737 (0.727)</td>
<td>0.871 (0.853)</td>
</tr>
<tr>
<td></td>
<td>No Workaround</td>
<td>3,497</td>
<td>0.842 (0.841)</td>
<td>0.957 (0.941)</td>
<td>0.896 (0.888)</td>
<td></td>
</tr>
<tr>
<td>$M_{\text{substituted}}$</td>
<td>Workaround</td>
<td>135</td>
<td>0.923 (0.744)</td>
<td>0.444 (0.319)</td>
<td>0.600 (0.446)</td>
<td>0.863 (0.858)</td>
</tr>
<tr>
<td></td>
<td>No Workaround</td>
<td>5,084</td>
<td>0.985 (0.982)</td>
<td>1.000 (0.997)</td>
<td>0.992 (0.990)</td>
<td></td>
</tr>
</tbody>
</table>

**Table 1. Predictive Performance per Model for our approach and the baseline in brackets.**

The highest $AUC_{\text{ROC}}$ can be observed for the $\text{Workaround}$ class $M_{\text{bypassed}}$. Also, for this class, the highest $\text{Precision}$ and $\text{F1-Score}$ could be achieved compared to the other models. The $\text{Support}$-ratio$^5$ of both classes is the highest for $M_{\text{repeated}}$, as there are most events in the class $\text{Workaround}$. In contrast, $M_{\text{added}}$ has the

$^3$https://data.4tu.nl/articles/dataset/Dataset_belonging_to_the_help_desk_log_of_an_Italian_Company/12675977.

$^4$We identified and labeled five of the seven workaround types proposed by Weinzierl et al. (2022) in the event log. Thus, only five types are considered in the demonstration.

$^5$We define the $\text{Support}$-ratio as: $\text{Support of class Workaround} / \text{Support of class No Workaround}$.
A Method for Workaround Prediction

Discussion and Future Research

While research on workarounds has recently gained interest in IS, much remains to be done. One major issue is the lack of data-driven methods to explore workarounds in a real-world context. We are the first to design a method for predicting workaround types in running instances of a business process. Our preliminary results indicate that, for one event log, our method predicted workarounds reliably.

A workaround prediction method, like the one described here, can have significant implications for research and management. From a theoretical perspective, our method can be considered the first to predict workarounds in real-time (Grisold et al. 2020). Based on the prediction, the impact of different types of workarounds on process change can be better understood, inducing an organizational learning process to improve business processes’ performance (Beerepoot et al. 2021). Related research emphasized that it is crucial to develop and implement (counter-)measures to mitigate the negative impacts and leverage the positive impacts that workarounds can exert on organizational structure (Laumer et al. 2017). Predicting workarounds emerging in business processes yields fundamental implications on the current conceptualization of workarounds in the IS literature, i.e., regarding the temporality of workarounds (Alter 2014). By predicting workarounds, their diffusion in an organization can be proactively managed and aligned with organizational goals. Hence, the risk that workarounds that have negative consequences diffuse uncontrolled in an organization can be minimized, whereas the institutionalization of positive workarounds can be promoted.

For process managers, identifying workarounds before they occur can provide massive benefits for actively managing (often hidden) process change by proactive workaround handling, while recognizing workarounds on an organizational level can currently take months or even years. Predicting workaround types, can outline novel ways for business process redesign (Dumas et al. 2018). This way, process redesign can be conceptualized as a dual innovation process, complementing traditional top-down change processes with bottom-up driven innovation processes.

In future research, we plan to finalize the ongoing iterative design and evaluation cycle. First, we will empirically evaluate the IT artifact’s power to predict workarounds in business processes. Our method learns DNN models with a BiLSTM layer. This type of DNN architecture has shown promising predictive performance in literature (e.g., Wang et al. 2019; Weinzierl et al. 2021) and our preliminary results. However, to assess the performance of our method’s prediction models, we will compare their results with other types of DNN models. Another aspect to consider is to frame the workaround prediction as a multi-classification problem, which we will also investigate in this step. Finally, as workarounds typically change over time (e.g., Weinzierl et al. 2022), we will also consider concept drift of the trained models applied to process-based event stream data. Second, we will test the method in a real-world setting to investigate its applicability and its contribution to process management. This approach is in line with the human risk & effectiveness evaluation strategy proposed by Venable et al. (2016), enabling us to apply our method to one or more event logs, comprising all seven workaround types conceptualized in Weinzierl et al. (2022). In a conclusive step, we then plan to develop design principles (Gregor et al. 2020) that prescribe how a class of methods for workaround prediction in business processes ought to be designed. Once we finalized evaluating our method, we will initiate a second design and evaluation cycle to extend the IT artifact from a predictive to a prescriptive method. This method will not only predict workarounds in running business processes, but will also prescribe measures on how a predicted workaround ought to be managed proactively.

Conclusion

Workarounds represent non-compliant behavior in business processes that employees implement to overcome inconsistencies between process specifications and practical requirements (Brander et al. 2011). Originally implemented as temporary first-order solutions to problems on an individual level (Azad and King 2012), they can also trigger secondary design processes that may result in processes drift, transforming organizations (Alter 2015; Pentland et al. 2020). Uncontrolled process change can yield positive or nega-
tive effects on an individual and an organizational level. As workarounds can be an important means for bottom-up process redesign, we designed an innovative DL-based method to predict workarounds in business processes. While the method represents a new class of IT artifacts, it can also have strong implications for actively managing process change in an organization.

References


