Inhalt

Abbildungsverzeichnis

Tabellenverzeichnis

1 Versionen

Version	Änderungen
V1	Formatvorlage erstellt
V2	Abstand nach Absätzen geändert, Verzeichnisse an Seitenränder angepasst, Schnellformatvorlagen „Tabelle“ und „Fußnote“ erstellt, Erläuterungen zu Seitenzahlen, Abschnittswechsel und Querverweisen eingefügt
V3	Silbentrennung eingeschaltet, APA-Link ersetzt, Erläuterungen zu Verzeichnissen, Absätzen, Gliederung, Seitenumfang

Tabelle 1: Versionen der Formatvorlage

2 Einleitung

2.1 Das Titelblatt

Im späteren Verlauf erhalten Sie ein Titelblatt spezifisch für ihren Studiengang. Dieses sollten Sie vor die Arbeit anfügen.

2.2 Formatvorlagen

Dieses Dokument enthält mehrere Schnellformatvorlagen. Verwenden Sie wenn möglich nur die Formatvorlagen um Ihnen unnötige Formatierungsprobleme zu ersparen. Dies ist insbesondere für die Formatierung von Überschriften wichtig, diese werden sonst nicht im Inhaltsverzeichnis aufgeführt.

2.3 Verzeichnisse

Die Arbeit sollte ein Inhaltsverzeichnis, ein Tabellen- und Abbildungsverzeichnis beinhalten – selbstverständlich nur, wenn Tabellen bzw. Abbildungen im Rahmen der Arbeit verwendet werden. Sofern Sie die Formatierung beibehalten, können Sie mit Microsoft Word automatisch ein Verzeichnis erstellen lassen. Dies ist auch in OpenOffice möglich. Sie sollten ein Abkürzungsverzeichnis anlegen, sofern Sie Abkürzungen verwenden, die nicht Teil des allgemeinen Sprachgebrauchs sind. Verwenden Sie für die Verzeichnisse jeweils eine eigene Seite.

Außerdem gehört in jede wissenschaftliche Arbeit ein Literaturverzeichnis. Dies steht in der Regel am Ende der Arbeit. Die einzige Ausnahme dessen ist die Verwendung eines Anhangs. Bitte achten Sie bei Referenzen im Text[1] aber auch im Literaturverzeichnis auf eine einheitliche Zitation und Formatierung (vgl 3.5).

[1] Hier steht eine Anmerkung.

3 Formatierung

3.1 Grundsätzliches

3.1.1 Schriftart und Schriftsatz

Dieser Text ist selbst mit dieser Dokumentvorlage geschrieben und kann in formaler Hinsicht als Muster verwendet werden. In jedem Fall sollten Sie mit Ihrem Betreuer absprechen, ob diese Vorlage verwendet werden kann oder ob Änderungen notwendig sind. Schriftart **Times New Roman** oder **Times** in Schriftgröße **11pt.** und Zeilenabstand **1,5**. Der Text ist in **Blocksatz** geschrieben.

3.1.2 Gliederung

Dies ist ein Beispiel für eine dritte Gliederungsebene. Sie sollten in Ihrer Arbeit nicht mehr als drei Gliederungsebenen verwenden. Unterkapitel sollten niemals alleine stehen, d.h. wenn Sie eine Unterteilung vornehmen, erstellen Sie immer mindestens zwei Unterkapitel. Die Unterkapitel sollten thematisch und vom Umfang her gleichwertig sind.

3.1.3 Absätze

Absätze sind dazu gedacht, unterschiedliche Gedankengänge räumlich voneinander zu trennen und die Lesbarkeit Ihrer Arbeit zu erhöhen. Sie sind nicht geeignet, den Seitenumfang Ihrer Arbeit künstlich aufzublähen. Zu viele Absätze stören außerdem den Lesefluss. Von daher sollten Absätze mindestens vier Zeilen umfassen, dürfen aber auch gerne länger sein.

3.2 Seite

Die Seitenränder lauten wie folgt:

- Oben: 4cm
- Links: 4cm
- Rechts: 4cm
- Unten: 3,5cm

3.3 Seitenzahlen

Die Textseiten Ihrer Arbeit sind mit arabischen Ziffern (1,2, …) zu nummerieren. Alle anderen Seiten (Inhalts-, Abbildungs-, Tabelle-, Literaturverzeichnisse, Anhang, etc.) sind mit römischen Ziffern (I, II, …) zu nummerieren. Dafür benötigen Sie in Word die Funktion des Abschnittswechsels, die in diesem Dokument bereits

vorhanden ist. Das Deckblatt zählt als erste Seite, die Seitenzahl wird aber nicht angegeben. Sobald Sie zusätzliche Seiten (wie etwa das Deckblatt) eingefügt haben, müssen Sie ggf. die Nummerierung des Literaturverzeichnisses anpassen.

3.4 Abbildungen und Tabellen

Abbildungen und Tabellen sind durchzunummerieren und mit einem Titel zu versehen (siehe Beispiel Abbildung 1). Außerdem sollten Sie im Text auf das Bild bzw. die Tabelle verweisen. Entweder durch eine Aussage wie bspw. „wie in der folgenden Abbildung zu sehen“ oder aber durch einen Querverweis (s. Abbildung 1). Verwenden Sie dazu die Word-Funktion „Verweise“ -> „Beschriftung einfügen“, damit Ihre Verzeichnisse ordnungsgemäß erstellt werden können. Bitte prüfen Sie die Querverweise vor der Abgabe noch einmal besonders sorgfältig, da sie durch Verschiebungen innerhalb des Dokuments schnell kaputtgehen.

Gleiches gilt für Tabellen:

	Spalte 1	**Spalte 2**	**Spalte 3**
Zeile 1	Text		
Zeile 2			

Tabelle 2: Beispieltabelle

Platzieren Sie Ihre Abbildungen und Tabellen möglichst nah an der Stelle, wo sie referenziert werden, aber gleichzeitig platzsparend. Wenn Sie nicht mehr auf die Seite mit der Referenz passen, platzieren Sie sie (wie hier) auf die direkt folgende Seite ganz nach oben und führen Sie den Text auf der vorherigen Seite fort, damit keine halbleeren Seiten entstehen.

Abbildungen und Tabellen zählen zum Seitenumfang Ihrer Arbeit. Sehr große Abbildungen und Tabellen können im Einzelfall in einen eventuellen Anhang ausgelagert werden, dies sollten Sie individuell mit Ihrem Betreuer besprechen.

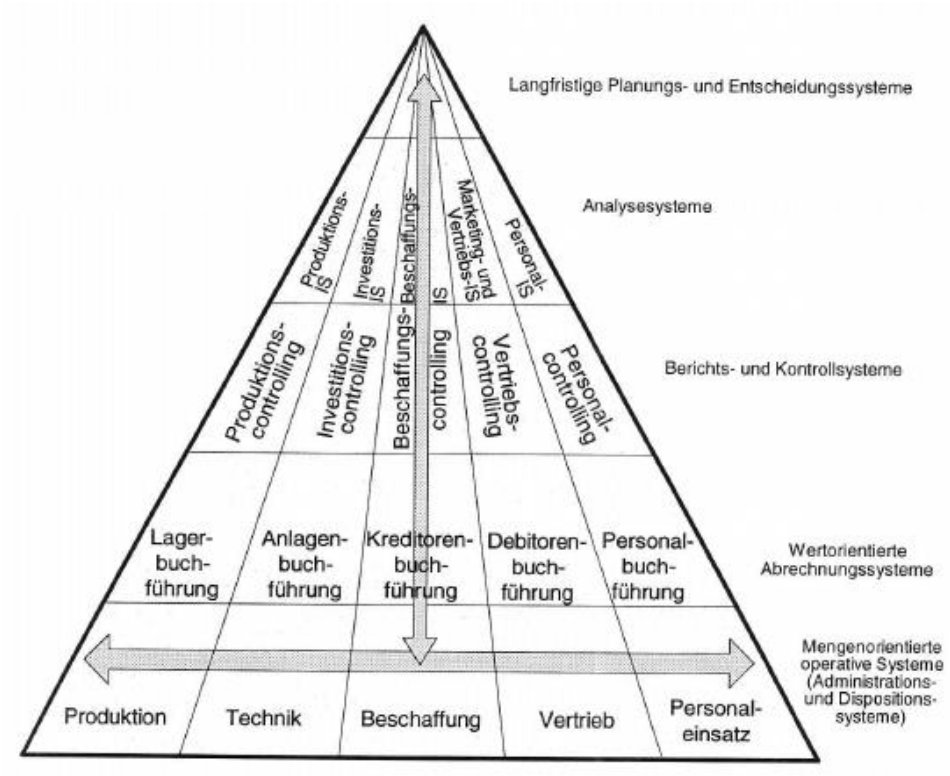


Abbildung 1: Informationssysteme nach Scheer(Scheer, 2008)

3.5 Zitation

Zur Zitation von Quellen verwenden Sie bitte den APA-Stil[2]. Beispiele für Journalartikel (Lasi, Kemper, Fettke, Feld, & Hoffmann, 2014), Artikel aus Konferenzbänden (Thaler, Ternis, Fettke, & Loos, 2015) oder Bücher (Scheer, 2008) finden Sie hier. Weitere Beispiele für unterschiedliche Arten der Literatur finden Sie unter anderem hier: https://onlinekurslabor.phil.uni-augsburg.de/course/text/3880/3998. Sie können zur Formatierung und Verwaltung der Literatur frei verfügbare Literaturverwaltungsprogramme wie Mendeley[3] oder Zotero[4] verwenden. Dort sind gängige Zitationsstile wie etwa APA bereits vordefiniert. Über ein Plugin können Sie die Zitationen direkt in Ihr Word-Dokument einfügen.

[2] http://www.apastyle.org/
[3] https://www.mendeley.com/
[4] https://www.zotero.org/

3.6 Fußnoten

Fußnoten sind nicht für Literaturangaben, sondern nur für zusätzliche Informationen oder Anmerkungen zu verwenden und mit dem Stil „Fußnote“ (Schriftgröße 10) zu formatieren.

4 Task and Goal

Process Prediction is a technique of Process Mining, that deals with the prediction of running (not finished) process instances. In the scope of this project general methods of process mining, especially the process prediction, should be implemented in Python and be visualized using a Dashboard. The created analysis should be evaluated using public datasets of the business process Intelligence Challenge (BPIC).

The focus of the dashboard is to do process discovery using techniques from process prediction, while also being able to do standard process prediction tasks. By creating a dashboard which lets the user choose actions from a given set of predictions, they can create a model of the possible future of a running process instance while also being able to create a universal model for the process instance.

To allow this, the created dashboard should have a graphic interface to display DFGs for simple and fast process discovery. To correctly apply the process prediction in this dashboard, the predicted actions for the process should be displayed in the dashboard and it should be able to select which predictions should be added in the process model. The progress of the model creation should also be shown using various metrics. When a model is completely created, it should be able to export it for use in internal tools.

The remainder of this documentation is structured as follows. First the topic and the state of science will be explained. Then my own contributions will be discussed and at the end all the results will be summarized.

5 Explanation of the topic

5.1 Introduction of the topic

Process Mining is a research discipline concerned with the analysis and discovery of processes. An example of such a process is the return of a defective product, starting with filing the claim and ending with the refunding of the price. The events and actions inside these processes can be collected as data and then stored in event logs. These can be used to get insights in the process. Process Mining can be separated into three different types, Discovery, Conformance checking and Extension (van der Aalst, 2010). This project consideres the discovery of processes. For this we use a discovery algorithm to convert the data given by the event logs to a process model.

Process Prediction is a Process Mining technique, which predicts the next actions of a running process instance. For this, the data from the event log can be used to calculate the probability of different actions following the ongoing process instance.

5.2 Motivation of the project and state of science

Process Mining uses many automatic algorithms to create process models from the event log. One prominent example is the alpha algorithm. Current process mining algorithms have many issues when creating the process from the event log. Many current algorithms tend to overfit or underfit to the given data. Underfitting refers to the problem of generalizing the process too much. The process model created by an underfitting process discovery technique would allow too many actions which are not actually possible in the process. Overfitting refers to the problem of creating too specific process models. The process model created by such a technique would be too restrictive and would not allow actions which are possible in the discovered process (van der Aalst, 2010).

Automated process mining techniques also only use the given data to create the process model, so the event logs to create the process model. Further insights given by experience or outside influences are not considered by these techniques. This problem can be solved by letting personal optimize the created process model by hand. But this approach is very time-consuming and requires experienced personnel.

These process mining techniques cannot be used to create decisions at runtime (Neu et al. 2022). A solution for this is the use of process prediction. Process Prediction techniques aim to get insight into the future of running process instances. To create the predictions, the data from an event log is used. First the prefixes should be extracted from the event log (Cevarolo et al. 2024). Prefixes are all sequences of actions recorded in a trace of the event log. So, prefixes are all traces and their respective subtraces in the event log. These prefixes are the basis for learning the predictive models (Cevarolo et al. 2024). Learning the prefixes can for example be done using deep learning methods (Neu et al. 2021).

5.3 Structure of the project work

The aim of this project is to combine process discovery techniques and process prediction techniques in a dashboard. Using these techniques allows different use cases of the dashboard created for this project. A completely new process model can be manually created using the predictions given by the given data. This manual creation of the process model is useful for inserting insights and experience of people into the process model. Also, overfitting and underfitting to the given data can be prevented, since the person creating the model can manually adjust the model while creating it. The process prediction functionality can also be used for the original use case of monitoring running process instances.

6 own contributions (with explanation of the current state)

6.1 Schedule and procedure

This project consisted of three main phases. In the first phase, I informed myself of the topic of process mining and it was planned what functions the dashboard should have. In the second phase, the dashboard has been coded, while adjusting the initial plan. Functions not considered in the first planning phase were added and other functions were not implemented in the final dashboard. One of the functions not implemented is the calculation of Generalization, Precision and Simplicity. Their calculation was too performance intensive to calculate it in real time, while their insights were not that important. Therefore, they were removed. In the last phase, the dashboard was tested, and bugs were removed.

6.2 Central results

The result is a process mining tool which is based on a prediction matrix. A dashboard can be used to discover a process using the predicted next actions in the process. This uses a matrix containing prefixes and the possible actions following these prefixes with their respective probabilities.

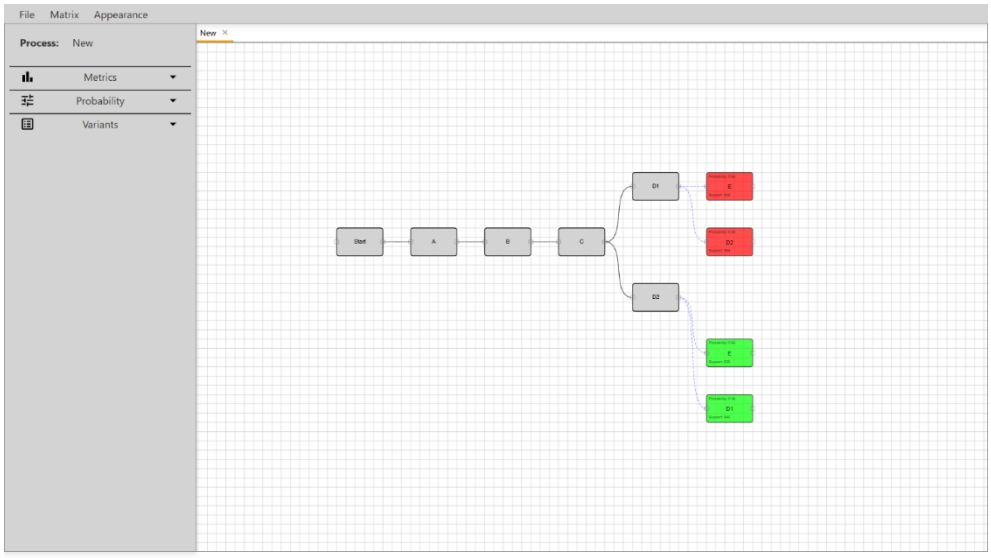
It runs as a website in the browser and most computations are done on the server. This ensures that the tool is scalable to weak hardware. On the front end only

computations considering the UI and the interpretation of the results of the backend computations are being handled.

6.2.1 Explanations of the Functions

The already discovered process model is displayed as a DFG. Activities are displayed as nodes in the DFG. The arcs between nodes should display, which activities can follow another. The predictions for following activities are being shown as slightly transparent, colorful nodes. The color of a prediction is determined by the predecessor of the prediction. This helps differentiate different groups of predictions. The calculation of how many predictions should be shown occurs either by an automated algorithm or is chosen by adjusting the minimal support and probability a prediction should have. This can be done using a slider. Furthermore, the probability and the support of the predicted activity are displayed on the node of the activity.

It is also possible to create arcs, without them being predicted by the matrix. For this, both the outgoing and the ingoing activity must be already discovered in the model, and we simply need to drag from the circle on the right of the activity node, we are starting from to the activity, the arc should end. This can be used to create actions sequences not possible by the predictions. This can give the user more freedom when creating the process model.



The DFG can be changed in multiple ways. Activities and arcs can be right clicked to open their respective pop-up menu. Here they can be deleted, and a comment can

be given for activities. The comment feature has a wide variety of use cases, like further describing an activity, for example if the activity name is misleading. Another use case can be to give comments about the activity in the process model itself, for example if the user wants to note that the activity could be omitted for the process to work. Deleting an activity automatically deletes all their incoming and outgoing arcs. Using a lasso tool by holding the right mouse button can be used to select multiple activities, which will be highlighted in light blue. Selected activities can be deleted all at once by deleting a single activity. When dragging the selected activities, all of them get moved. This can be used if the user wants to move a large batch of activities without having to move each one individually. Navigating in the graph display can be done by dragging the background using the left mouse button. Zooming is possible by using the scroll wheel.

Multiple models can be worked on at the same time using the tabs. When staring at the site, there is already a tab open by default. Each tab can be closed using the x symbol and when closing the last open tab, a completely new tab gets created automatically. In the file button in the header, entirely new tabs can be opened, or already saved models can be loaded from a file. Furthermore, the open model can be saved as a Json file, which can be loaded later. The tabs can be used if the user wants to compare multiple models of the same process, by testing multiple deviations, when choosing the predicted actions.

In the sidebar on the left, there is the option to name the current process model. There are also three dropdown menus containing different functionalities. The first dropdown menu shows performance metrics, like variant coverage, log coverage and fitness. Fitness can only be displayed for process models where the process log also exists. In the current implementation this only exists for the PDC_2020... matrix. In the second dropdown menu the minimum probability and support can be adjusted. By default, the algorithm to calculate the number of displayed nodes and therefore the support and the probability is used. But if it is needed to manually adjust the number of displayed predictions, then it can be disabled using the checkbox. When enabled, we use the sliders for support and probability to change the number of predictions displayed. This is used to control the number of predictions shown, by only choosing predictions whose support or probability is larger or equal than the selected values. Therefore, a smaller support and probability value lead to more predictions being shown. In the third dropdown menu, all variants in the matrix are being

displayed with their support. There are options to search for variants and to sort them by support. Searching for variants works by giving the name of the actions included in a variant. When inserting multiple names divided by commas of spaces, then only variants including all the given actions are shown in the list. If a variant is already completely covered, then it is displayed in a different color.

In the header there is a “File”, a “Matrix” and an “Appearance” button. These open further dropdown menus. The file dropdown menu has the opening, storing and loading features as mentioned above. But it also contains the function to export the discovered process as a petri net. Here the petri net can be exported either as a picture or as a pnml file. The matrix dropdown menu has the functionality of changing between the different available matrices. There can also be more matrices uploaded to discover different processes. These get stored as cookies in the browser. It is also possible to delete uploaded matrices if they are not being used anymore. In the “Appearance” dropdown menu there are multiple settings for the appearance of the site and the graph. Here the grid in the DFG display can be disabled and the feature to give predictions a different color can be disabled. Furthermore, there is the functionality to auto position the graph. This can be used to reposition the activities in very large graphs such that they are more readable.

6.2.2 Workflow example

When trying to discover the process model from a given matrix, the user starts by choosing the matrix they want to use. The user can upload a matrix or use one of the predefined matrices. Then they can see the starting activity on the dashboard. From this there should be multiple outgoing predictions displayed. Based on the support and the probability, the user can now choose which activity should be happening next, by clicking the prediction. If the user is already familiar with the process, he can also choose the following activity based on prior knowledge and not only based on the support and probability values. This can be repeated until the user is content with the created process.

Metrics and the variants displayed in the sidebar can be used to give an overview over how far the process discovery has advanced. Based on the variants we can see whether a possible, important sequence of actions could have been ignored. The feature to open multiple tabs allows to create multiple process models using the same

matrix, such that these models can be easily compared to each other. This can help to get a better understanding of the discovered model and the underlying process. Matrices can also be changed while working on the process model. This can be used, if multiple matrices of similar or the same process are available. Then the discovered model can be further expanded by activities from the other model, or simply the metrics of the different matrix can be displayed to check if the model is still applicable on different matrices (to fix overfitting for example).

If the process model is completely discovered, it can be exported as a petri net. It is possible to export it as a petri net picture. This can be used, if easy and fast access to the model is needed. We can also export it as a pnml file, such that it can be easily opened in other process mining tools, like pm4py applications. There it can be further analyzed and further metircs like fitness, generalisation, precision and [] can be viewed.

6.2.3 Used algorithms

4.2.3.1 Choosing predictions

First all prefixes in the matrix are being checked, to see if they are possible in the model discovered. We do this by iterating over each action in the prefix and checking whether it is possible in the currently discovered model. If a prefix is possible, we return all the predictions with a support and probability value, that is higher than the given minimum support and probability value. When using the auto probability algorithm, the check for the support and probability is omitted and which predictions to show is done later.

The predicted actions are grouped by the predecessor action, since multiple prefixes can lead to the same prediction. This can happen if there are multiple paths inside the process model that lead to the same action. The support for the grouped prediction is being calculated by adding all the support values and the probability is being calculated as the weighted average of all the probabilities. Here the weight for the weighted average calculation is the support of each probability in the group.

After that we have all the predictions possible in the current model. Now we need to check if the prediction needs to be added. If it already exists in the discovered model, it does not need to be added. So, if there exists an arc from the predecessor action to

the action with the same name as the prediction. Otherwise, the prediction will be added.

4.2.3.2. Selecting a prediction

When clicking on a predicted action (on the node), this prediction should be added to the discovered model. For this, we first check if an action with the same name already exists. If not, then we first must add the action to the discovered model. After that, we add the arc from the predecessor of the prediction to the action.

4.2.3.3. Auto support algorithm

The auto support algorithm calculates the number of displayed nodes automatically. This ensures that the number of displayed nodes always stays low to an easily readable graph. For this first the number of predictions should be calculated and then we get all the predictions that should be displayed by calculating the minimum support indirectly.

The maximum number of predicted actions is calculated as a function of the already discovered actions:

f(x) = 2·ln(x)2+3fx = 2·lnf_0x2+3

This ensures that the maximum number of predicted nodes grows fast for few actions in the DFG, while decreasing the growth for high number of nodes. The function assures that the maximum is never smaller than 3. This calculated maximum number is used to ensure that the number of predicted nodes does not get too large, such that the predictions can be assessed easily. If less predictions are given by the matrix, than this calculated number, then this amount will be displayed.

The next step is to get the predictions as described in 4.2.3.1. These are returned as a dictionary with their support. Now we must reduce the number of displayed predictions to the calculated numbers of displayed prediction nodes, while only showing predictions with the highest support. For this we first order a copied list of the support values of each prediction descending by support. If the calculated maximum number of predictions is n, then we output the n-th support value. Since the

predictions are returned as a dictionary, we cannot sort the dictionary itself and therefore we cannot extract the predictions directly from the ordered list. Therefore, we must check for each prediction in the dictionary, if its support is larger than the calculated minimum support.

After that, to further increase the readability of the graph, we ensure that after each already discovered action in the model, there are a maximum of three predictions following. For this also the nodes with the lowest support are removed.

6.2.4 Used tools and frameworks

The frontend of the site has been coded in Typescript using React and Vite. API calls are handled by Axios. The work area, where the process model is displayed as a DFG is handled using the Conva.js library.

The backend is done in Python using flask. The conversion to a petri net and the fitness calculation is handled by pm4py and the creation of the picture of the petri net is done using graphviz. The matrices from the csv files are opened using pandas data frames.

6.3 Discussion of the results

7 summary

8 Literatur

Lasi, H., Kemper, H.-G., Fettke, P., Feld, T., & Hoffmann, M. (2014). Industrie 4.0. *Wirtschaftsinformatik, 56*(4), 261-264.

Scheer, A.-W. (2008). *Wirtschaftsinformatik Studienausgabe - Referenzmodelle für industrielle Geschäftsprozesse* (Vol. 2.). Berlin, Heidelberg: Springer Verlag.

Thaler, T., Ternis, S., Fettke, P., & Loos, P. (2015). *A Comparative Analysis of Process Instance Cluster Techniques*. Paper presented at the 12th International Conference on Wirtschaftsinformatik, Osnabrück, Germany.

Eidesstattliche Erklärung

Hiermit erkläre ich, dass ich die Arbeit mit dem Titel

_____

eigenständig erbracht, keine anderen als die angegebenen Quellen und Hilfsmittel benutzt und die aus fremden Quellen direkt oder indirekt übernommenen Gedanken (Texte, Textbausteine und/oder -fragmente) als solche kenntlich gemacht habe. Die Arbeit wurde nicht, auch nicht in Teilen, unter Verwendung eines textbasierten Dialogsystems (wie ChatGPT oder andere Werkzeuge basierend auf Large Language Models) oder auf andere Weise mit Hilfe einer künstlichen Intelligenz von mir verfasst. Die Arbeit habe ich in gleicher oder ähnlicher Form oder auszugsweise noch keiner Prüfungsbehörde zu Prüfungszwecken vorgelegt. Des Weiteren bestätige ich, dass die schriftliche und die elektronische Version der Arbeit identisch sind.

Mir ist bekannt, dass Zuwiderhandlungen gegen den Inhalt dieser Erklärung einen Täuschungsversuch darstellen, der grundsätzlich das Nichtbestehen der Prüfung zur Folge hat.

_____, den _____

Unterschrift