An Interactive Layout Tool for BPMN

Philip Effinger, Martin Siebenhaller, Michael Kaufmann
Wilhelm-Schickard-Institut für Informatik
Eberhard-Karls-Universität, Tübingen, Germany
Email: {effinger,siebenha,mk}@informatik.uni-tuebingen.de

Abstract—Business processes are at the core of today's business world. Most of the effort put into business processes in practice is either the task of designing a new process or the task of analyzing and improving an existing process. In both cases, visualizations of the process models support the user in achieving his objectives.
In this paper, we present the first prototype of our tool "BPMN-Layouter". Our supported notation language is the business process modeling notation (BPMN). The tool was implemented to integrate several concepts that we developed and adopted for the use in BPMN modeling. The integrated approaches comprehend the tasks of finding an automatic layout for a BPMN model, increasing the readability of given visualizations and developing further existing BPMN models in an interactive fashion. Also, a method for automatic division of complex BPMN models is provided, too.

Keywords—BPMN, BPMN Layout, BPMN Visualization, Graph Drawing

1 Introduction

BPMN becomes more and more popular documented by the OMG strongly working towards specification of version 2.0. The acceptance is also shown by a growing integration of BPMN into common modeling software tools. The current version 1.2 was passed in January 2009 and can be found in [1]. A short introduction for BPMN is offered in [2]. BPMN provides a set of notations for modeling business processes. Among the most important elements are flow objects, e.g. activities, events and gateways. The flow objects are connected by connecting objects, e.g. sequence flows or message flows. Additionally, BPMN models are structured by assigning flow objects to swimlanes and by adding artifacts, e.g. annotations, to connecting objects or flow objects.
However, the techniques for visualization used by common modeling tools could not keep pace with the demands required by a complex modeling notation as BPMN stands for. The demands are numerous and must be considered by the visualization, e.g. BPMN allows the definition of swimlanes and pools or flow objects can build a group. An automatic layout can help to avoid mistakes during the process modeling, e.g. it supports the communication among designers and helps to understand unknown models. On the other hand, 'bad' layouts of models can cause misunderstandings among model designers and they can be the cause for erroneous models because of the models' complexity.
The BPMN modeling tools are often limited to support only the design and not the layout of models. Computer-assisted layout techniques are rarely spread among the tools although they are very helpful when models become complex. Some modeling tools offer a simplistic layout concept provided to an user. Unfortunately, until today, limitations and shortcomings in these layouts appear on regular basis and could not yet be overcome. The shortcomings of BPMN layouts become clearly visible when using layout mechanisms provided by the tools as presented in Section 3.
The aim of this paper is to present several methods capable of overcoming existing shortcomings of modeling tools and demonstrate our obtained results. Our concepts are based on graph drawing methods which are adopted for modeling BPMN. In Section 2, we elaborate on criteria and requirements for 'good' layouts. Thereafter, we shortly introduce our layout method in Section 4.2.1, followed by two improvements: the insertion of links for long connecting objects is presented in Section 4.3 and the division of complex process visualizations follows in Section 4.4. In Section 4.2.2, our approach towards interactivity is presented. Before we conclude, Section 4.5 demonstrates our implemented tool.

2 Requirements of a 'Good' layout

2.1 Aesthetics - Measurements for a layout

An aesthetics measures a graphical property of a drawing that should be optimized to increase readability. Unfortunately, up to now there are neither research work nor empirical studies for aesthetics of BPMN diagrams. However, research was performed in the area of aesthetics of UML diagrams [3] or general explorations for diagram aesthetics [4], but none of them is applicable without extension to BPMN diagrams. Since the underlying structure of BPMN diagrams are graphs, we will first discuss aesthetics that apply to abstract graphs (graphs without special semantics).
An overview of aesthetic criteria applied to drawings of abstract graphs is given in [5], [6]. The most common are:

- Minimize the number of crossings of connecting objects (CROSSING).

- Minimize the area of the drawing (AREA).
- Minimize the number of bends of connecting objects (BEND).
- Maximize the smallest angle between two connecting objects incident on the same flow object (ANGLE).
- Minimize the number of overlapping flow objects and connecting objects (OVERLAP).
- Maximize the number of orthogonal connecting objects (connecting objects that are drawn as a sequence of horizontal and vertical line segments) (ORTHOGONAL).
- Maximize the number of connecting objects respecting workflow direction (connecting objects that are drawn monotonically in a prescribed direction) (FLOW).
- Maximize symmetry (SYMMETRY).

Note, that there are some contradicting aesthetics, e.g. FLOW and AREA. The drawing of a graph with a given set of aesthetic criteria can be seen as a multi-objective optimization problem. The set of applied criteria depends on the semantics of the graph as well as on individual user preferences.

The effect of aesthetics BEND, CROSSING, ANGLE, ORTHOGONAL and SYMMETRY on the readability of drawings were analyzed empirically in [3]. There, CROSSING was found out to be the most important, followed by the less important aesthetics BEND and SYMMETRY. ORTHOGONAL and ANGLE had no significant effects.

2.2 Layout Specifics of BPMN

In general, the visual notation of BPMN diagrams leads to the following layout requirements:

- Flow objects have different sizes (ELEMENT_SIZE).
- We have to consider partitions, e.g. (collapsed/expanded) pools and swimlanes (PARTITION).
- Subprocesses may be nested and connecting objects can start/end at a subprocess. This corresponds to the concept of graph clustering (CLUSTER).
- Handle annotations that are attached to connecting objects (ANNOTATION).
- Handle labels of pools, swimlanes, flow objects and connecting objects (LABEL).

Since BPMN diagrams are based on a graphtheoretic concept, we are able to employ graph drawing techniques to draw BPMN diagrams. There is a variety of graph drawing algorithms taking different aesthetics and requirements into account [5], [7]. There are also approaches for handling labels and clusters. Note, that in graph theory the flow objects are usually called vertices and the connecting objects are called edges. However, in this work we use the terms flow objects and connecting objects respectively to be consistent with the BPMN terminology.

2.3 Standards for Designing BPMN Diagrams

Common accepted standards for the design of BPMN diagrams, as derived for activity diagrams in [8], are not yet defined. Conventions are given for flow objects in the specification, but they are not directly applicable for the diagrams as a drawing.

However, considering known diagram types, e.g. UML activity diagrams or network diagrams [3], [8], [9], principles for BPMN diagrams can be given by:

- Minimize the number of crossings of connecting object (CROSSING).
- Draw connecting objects orthogonal (ORTHOGONAL).
- Flow objects/connecting objects are not allowed to overlap (ELEMENT_SIZE, OVERLAP).
- Incoming and outgoing connecting objects enter a gateway on distinct sides. If a gateway has two outgoing connecting objects they should be connected at opposite sides (BIMODALITY).

In practice, BPMN diagrams contain much less flow objects and connecting objects compared to large network graphs with millions of nodes. This fact allows us to use more complex algorithms for BPMN diagrams.

Since BPMN diagrams use sequence and message flows it seems to be natural to include aesthetics FLOW. In BPMN diagrams the flow direction is usually top-to-bottom or left-to-right. OVERLAP is also important; especially overlaps between flow objects should be avoided. SYMMETRY seems to be useless to BPMN diagrams because they do not have a symmetric structure.

3 Related Work

This section will give a summary on existing conceptual work towards automatic layout techniques for BPMN and provide a short summary on existing commercial modeling tools for BPMN. The number of existing (commercial) solutions for automatic layouts for BPMN is very low. A selection of popular tools for BPMN modeling and their layout capability is presented in Section 3.2.

3.1 Related Conceptual Work

Up to now, there is no relevant work on layout concepts for BPMN diagrams. However, there is major work on related layout algorithms for software structures, e.g. UML class diagrams or UML activity diagrams.

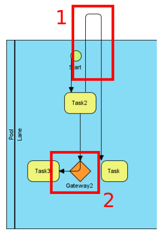
Representatives of two different approaches for UML diagrams are UML-Kandinsky [10], [11] and SugiBib [12]. The first one is based on refinements of the fundamental topology-shape-metrics (TSM-) approach which has been developed for orthogonal layouts. The second one uses the well-known concept of Sugiyama for layered layouts. They both support aesthetic criteria such as FLOW, ORTHOGONAL, OVERLAP, CROSSING in various ways. SugiBib is also able to consider advanced properties like

CLUSTER but on the other hand it is rather weak for basic properties like ELEMENT_SIZE or ORTHOGONAL. None of the above stated approaches except [11] is able to handle PARTITION.
There is also some early work on the visualization of general process diagrams. Process Diagrams are related to flowcharts and visualize the workflow in a process or system. The layout algorithms described in [13] and [14] support FLOW and PARTITION. The second one is also able to support ORTHOGONAL, CROSSING, BEND and OVERLAP. However, both solutions do not include CLUSTER. Moreover, their visual results are mostly overcome today.

3.2 BPMN Modeling Tools

As we have seen there is no conceptual approach covering all layout requirements for BPMN diagrams. On the other side there exist quite a few commercial products that claim to support BPMN diagrams.
In the following subsection we review some popular tools:

1) *Business Performance Edition* of ARIS-Suite developed by IDS Scheer:
(http://www.ids-scheer.com/en/ARIS/)
In the *Business Performance Edition* (BPE), ARIS contains *Business Architect* in version 7.1 which offers modeling in BPMN 1.0. In [15], functionality of *Business Architect* concerning BPMN-modeling is described. The focus of BPE is on syntax validation and simulation of models. During modeling in *Business Architect*, an automatic resizing of pools and swimlanes is offered. Further layout techniques are not known.
2) *Business Process Visual Architect 2.4* by Visual Paradigm:
(http://www.visual-paradigm.com/product/bpva/)
Business Process Visual Architect (BP-VA) is developed for modeling with BPMN and related workflow notations. Its appearance and user interface is mature. BP-VA offers several alternatives for producing a layout of a model. However, all but one are standard layout algorithm for general graphs and are not adapted to specifics of BPMN. The remaining layout technique produces non-satisfactory layouts. We will document the shortcomings by two examples where the layout problems are numerous, see Figure 1.
3) *WebSphere Business Modeler Advanced 6.2* by IBM:
(http://www-01.ibm.com/software/integration/)
WebSphere Business Modeler Advanced 6.2 (WBMA) offers modeling in BPMN-style. The BPMN standard is not fully supported but rather a subset of flow objects is available. The automatic layout technique is limited to moving flow objects in direction of FLOW (fixed from left-to-right) in order to avoid overlapping flow objects (OVERLAP).



(a) Example produced by the BP-VA: Connecting object routing is hardly comprehensible since the upper connecting object route (1) leaves the pool area and the inner connecting object (2) overlaps the gateway object.

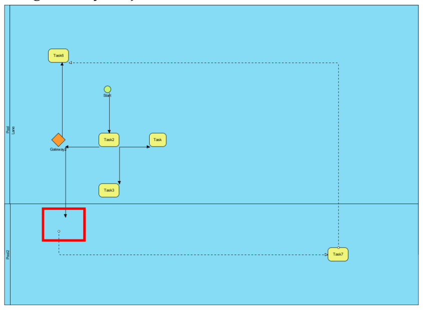

(b) Layout of BP-VA. A flow object is covered by a pool; only connecting objects mark the flow object's existence.

Figure 1: Examples for bad routing (a) and the layout produced by BP-VA for a model with two pools (b).

Rerouting of connecting objects is not available, crossings of connecting objects are not removed. However, a compaction can be used to reduce the area of the diagram (AREA). The compaction shortens connecting objects to reduce area size, but the topology of the diagram is not changed and, again, connecting objects are not rerouted. An example for the layout result of WBMA is depicted in Figure 2.
Aside of connecting object routing in visualizations, support of pools and swimlanes is a challenge only a few visualizations are capable to accept. Many tools, e.g. WBMA, simply visualize the process' workflow.

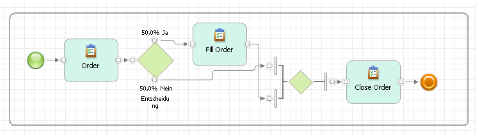


Figure 2: IBM's *WebSphere Business Modeler Advanced 6.2*. Flow object positioning is computer-assisted but only in flow direction.

In conclusion to the comparison of used layout techniques in popular software tools for BPMN modeling, layout of BPMN models has been neglected in the past and often does not match the requirements for layouts as explored in Section 2. Worse, in numerous cases, a BPMN model was changed or rendered invalid by layout techniques. This must be avoided and is not acceptable for the user.

4 Tool Design and Visualization

4.1 Features

Our implementation was integrated into a tool called "BPMN-Layouter". A screenshot is shown in Figure 3. For implementation purposes, we used the *yFiles* graph library, [16].

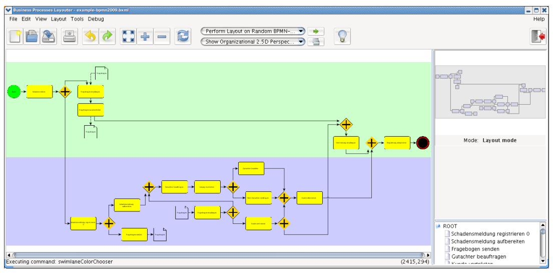

Business Processes Layouter - example-bpmn2009.bxml
File Edit View Layout Tools Debug
Help
Perform Layout on Random BPMN-...
Show Organizational 2.5D Perspec...
Mode: Layout mode
ROOT
Schadensmeldung registrieren 0
Schadensmeldung aufbereiten
Fragebogen senden
Gutachter beauftragen
Executing command: swimlaneColorChooser
(2415,294)

Figure 3: Screenshot of our JAVA-based tool for design and layout of BPMN models.

Supported features of the tool are among others:

- *Automatic layouts:* Among the standard layout techniques, e.g. hierarchical, orthogonal or organic layout, the most interesting layout technique provided by our tool is the automatic layout algorithm which we will present in the next Section 4.2.1. The algorithm is adapted to BPMN, e.g. it respects amongst others: swimlanes (PARTITION), distinct sides of in-/outgoing connecting objects at gateways (BIMODALITY), general flow direction (FLOW) and orthogonal connecting object routing (ORTHOGONAL). In our prototype, nested subprocesses are handled as one flow object. This allows the layout of subprocesses in the whole model, but yet prevents the layout of objects in the subprocess. Also, attached flow objects are yet treated as adjacent objects. These limitations are addressed in [17] and will soon be integrated into our prototype.
- *Model design:* Designing a model from scratch is supported by an easy-to-use 'edit-mode'. With a simple click on the pane a dialog pops up where the user provides necessary attributes for the new flow object, e.g. flow object type, swimlane assignment, name or a comment for process documentation. Connecting objects are drawn by drag-and-drop from source to destination flow object. An option dialog for connecting objects allows settings of flow direction and flow type. In a sidebar, attributes of flow objects or connecting objects can be changed at any time.
- *Divisions and Links:* The tool integrates our approaches for divisions and the insertion of links for BPMN diagrams presented in the upcoming sections 4.4 and 4.3. From an inserted link, the view can automatically be focused to the associated link. Also, moving the mouse on one of two links representing a substituted connecting object, the substituted connecting object is temporarily faded in. If the user wishes to inspect both links in the view at a time, the zoom is automatically adapted and the view is focused such that both links are visible.
- *Free Navigation and Zoom:* The user can move freely on the model. Mouse supported navigation includes infinitely variable zoom levels. The zoom level can also automatically be fitted to inspect the whole model. In the sidebar of our tool, an overview pane of the whole model shows the surroundings of the current displayed part of the model. Supporting free navigation, complex models can easily be handled and inspected.
- *Import and Export:* Although many BPM tools often support XPDL as interchange format [18], there is not yet a standard serialization format for model exchange for BPMN. Our prototype provides a variety of export formats, both graphical or data-oriented. Graphical formats can be chosen among the formats of images for JPEG, PNG or SVG. Data formats are either graph-based (**.ygf*, **.gml*) or based on XML (**.bxml*). We developed the file format (**.bxml*) for support of BPMN specific attributes. It is an extension of GraphML [19]. With version 2.0 of the BPMN specification, a standard exchange format is planned to provide which will then also be integrated into our tool. Until then, the import function is yet limited to the graph-based formats (**.ygf*, **.gml*) or our BPMN-specific file format **.bxml*.
- *Undo/Redo-Feature:* An undo-/redo-feature is also included. It allows to compare different layouts produced for BPMN models. Also, if a certain layout is unsatisfactory for the user, it's easy to undo the last steps and try different parameters, e.g. when performing divisions, see Section 4.4.

4.2 Layout Algorithms

4.2.1 Algorithm for Automatic BPMN-Layout

As mentioned in Section 1, BPMN diagrams are based on a graphtheoretic concept. Consequently, we need to define the graph for modeling an BPMN model:

Definition 1 (BPMN-graph) *A BPMN-graph is a graph $G = (V, E)$ with the following additional information:*

- *A mapping $flow\_object\_type : V \rightarrow T$, where T denotes the set of types of flow objects. Each flow object $n \in V$ is mapped to exactly one type $t \in T$.*
- *A mapping $connecting\_object\_type : E \rightarrow C$, where C denotes the set of connecting object types in BPMN. Each connecting object $e \in E$ is assigned to exactly one connecting object type $c \in C$.*

An automatic layout approach for BPMN-graphs has to support the specific requirements given by the BPMN notation as introduced in Section 2. Since BPMN-graphs are usually drawn using orthogonal routes for connecting objects, we use an orthogonal layout approach for calculating the initial layout of a given BPMN-graph.

More precisely, our approach employs the implementation described in [11] that incorporates different constraints needed for the automatic layout of activity diagrams which are related to business process diagrams. The supported constraints include partitions (a generalization of swimlanes), clusters (subprocesses/groups) as well as a common workflow direction of connecting objects which is especially important for such diagrams. Used techniques are based on Sugiyama's algorithm [20] and the *Topology-Shape-Metrics* (TSM) approach [21]. Thus, all layout requirements demanded by BPMN-models are fulfilled. Further details of this implementation can be found in [22], [23].

4.2.2 Interactive Layout

In the practice of modeling, changes must be done rather often in an existing model. The same applies for layouts; a layout can undergo changes since the underlying model changes. However, if the model changes, the layout should not be changed substantially. Thus, the goal is not to destroy the user's mental map of a model. The *Sketch-Driven-Layout*-approach (SDL) addresses this challenge. The original idea of SDL stems from [24]. SDL considers an existing drawing and calculates a new layout targeting to fulfill the layout criteria as listed in Section 2. It is built on a Bayesian framework in order to measure the amount of changes in a graph, or, more precisely, changes in connecting object bends and angles when performing a layout algorithm. Note that the topology of the model is preserved. The new layout is found by solving an optimization problem with a weight function given by the changes in connecting object bends and angles.

However, the original idea was limited, e.g. since PARTITION was not considered. Thus, we had to extend SDL such that it considers swimlanes beyond preserving the user's mental map. This could be solved by extending the weight function of the optimization problem, see the extension and implementation of SDL adapted for BPMN in [22].

The benefits of SDL are enormous: Any time, a BPMN model is changed, an automatic layout for the model can be calculated while changing the mental map as little as possible. This allows to integrate the layout step in the process of designing the model: When a connecting object or a flow object is added/removed the SDL is called and adapts the layout to fulfill the layout requirements. Using SDL for interactive layout enables us to offer an automatic and interactive layout approach that supports the human designer during the design process of a model.

4.3 Automatic Insertion of Links

In this section, we propose an approach that allows automatic insertion of links in BPMN layouts in order to avoid unnecessary long connecting object routes. Thereby, a high density of the layout, caused by many connecting objects in complex models, is reduced. The main issue is how to define the badness score of a connecting object depending on its routing. In our approach, this is done by defining a score for every connecting object representing its badness value and thereafter replace the connecting object with the highest (and therefore worst) badness value.

The principle of off-page connectors is used to replace connecting objects by a pair of placeholders, so-called links, which are connected to the corresponding endpoints of the replaced connecting object. Links belonging to the same connecting object get the same unique label to denote their correlation. Links offer a way to reduce the number of unaesthetic routes of connecting objects, i.e. connecting objects that contain many bends and/or crossings with other connecting objects. Note, that the connecting objects incident to links can always be drawn in without crossings and with at most one bend.

Candidates for a replacement by links are determined by means of a badness score function.

Definition 2 (Badness Score of a Connecting Object) *The badness value $bad : E \to \mathbb{N}$ of a connecting object $e \in E$ in an orthogonal drawing of a BPMN-graph $G = (V, E)$ is a score that is given by*

$$bad(e) = \alpha \cdot \#crossings + \beta \cdot \#bends + \gamma \cdot length$$

where $\alpha, \beta, \gamma \in \mathbb{N}$ are constant weights for the factors of the number of crossings ($\#crossings$), bends ($\#bends$) and the sum of the length of the connecting object's segments ($length$).

We iteratively choose the connecting object $e = (v, w)$ with maximum badness value, where v and w are the connected flow objects. After removing e from the model, we insert two links l_e^1 and l_e^2 as well as the connecting objects (v, l_e^1) and (l_e^2, w). Furthermore, we have to update the badness values of the remaining connecting objects that may have intersected with connecting object e.

The user can control how many links are to be inserted, e.g. this can be preset by a given minimum score bound or a maximum fraction of the number of connecting objects.

Additionally, the given badness score function can easily be extended in order to consider semantic requirements of BPMN, e.g. towards an exclusion of sequence connecting objects that should not be replaced in any case.

Therefore, our approach is an extensible solution for reducing the density in complex BPMN models. Note, however, that all information of the BPMN model is kept since any inserted link is labeled to identify the corresponding link.

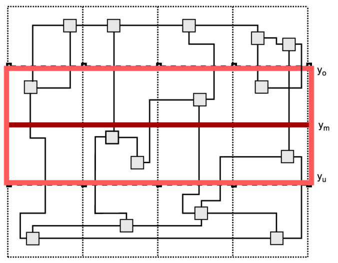
4.4 Divisions

In cases where process models become very complex and diagrams become large, it is desirable to divide the resulting diagram into smaller pieces, e.g. for printing a diagram on several sheets of standard size paper. The study in [25] showed that subprocesses are prefered to complete models in large diagrams. Although, divisions on diagrams do not correspond to subprocesses in models, the approach uses the concept of subdividing a model. In the following, we give an algorithm that divides BPMN-graphs automatically, subject to constraints, e.g. the size of sheets the BPMN-graph is to be printed on or the number of resulting pieces.

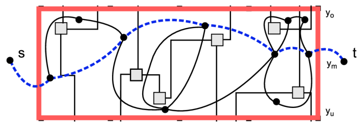
In the following definition, we introduce constraints for a division, e.g. constraints may comprise requirements concerning partitioning of a BPMN-graph or maximum area size for a BPMN-graph.

Definition 3 (Division of BPMN-Graphs) *A* Division *of a BPMN-graph $G = (V, E)$ with given constraints C partitions G into sets of flow objects $V_1, \ldots, V_k$ with $k \geq 2$ such that $V_i \cap V_j = \emptyset\ \forall i \neq j$. The subgraphs induced by V_i on G have to satisfy C. Connecting objects of $A = \{(v, w) \in E \mid v \in V_i, w \in V_j$ and $i \neq j\}$ are called* division connections.

In BPMN-graphs, the aim of a division is to minimize $|A|$ and to obtain subgraphs of nearly equivalent size in terms of flow objects or area size.



(a) Placing a horizontal center band on the underlying graph. The red box denotes the center band. Swimlanes are depicted by dotted lines.



(b) Determining a division using the dual graph. The dashed blue edges denote a shortest path from s to t and thus induce a cut with a minimum number of split edges in the center band.

Figure 4: Determining a cut in a graph.

In order to find appropriate routes, we introduce the new idea of a *center band*. The *center band* is a rectangular space in the graph through which a cut runs, see Figure 4(a). Depending on user's preferences, the *center band* can be given as user input or it can be set automatically to a default fraction of the graph's size.

We now describe how to perform a horizontal cut, a vertical cut is performed analogously. For a horizontal cut the width of the center band is set to the width of the bounding box surrounding the graph. The height of the center band is set to a predefined value $y_o - y_u$ around the midpoint center y_m of the *center band*, see Figure 4(a). The predefined value, if not given by the user, is preset with a default fraction of the graph's height. In our evaluations, we found that a value that corresponds to 10% of the graph's height is a good choice for most graphs.

The core of the division algorithm is a dual graph routing inside the *center band* using Dijkstra's shortest path computation [26]. The dual graph $G'_D = (V'_D, E'_D)$ is constructed on the *cut graph* $G' = (V', E') \subseteq G$ induced by the *center band*. Further details of the dual graph routing in our division algorithm can be found in [27].

Since there is a one-to-one relation between connecting objects of E' and edges of E'_D, it is easy to set higher weights for specific connecting objects of E' that should not become division connections. Those specific weights can be set by the user and they can be set to comply with BPMN specific semantic preferences, e.g. a data object assigned to a connecting object should not be cut. The connecting object weights are then passed to the corresponding edges of E'_D.

An example of a horizontal cut can be found in Figure 4(b). Analogously to horizonal cuts, a vertical cut is performed by using a vertical *center band*, see the example in Section 4.5.

Performing a shortest path computation on the dual graph, we obtain the division connections for the original graph. Those connecting objects have to be removed in order to split the graph. However, a connecting object removal causes information loss. Thus, we insert two replacement objects (links) for each such connecting object, analogously to the insertion of two links in Section 4.3. Examples for the insertion of replacement objects as placeholders can be inspected in Section 4.5.

After performing a cut, the computed subgraphs are relayouted using the sketch-driven approach as presented in Section 4.2.2.

4.5 Demo

In this section we demonstrate the results of our visualization and the improvements on a real-world example. In the example case, we applied our software to the input graph shown in Fig. 5(a) which was drawn with the help of our tool. The graph considers the workflow in left-to-right perspective. The core layout produced the drawing given in Figure 5(b). The workflow is preserved in the left-to-right perspective. All connecting objects have orthogonal routes.

Assuming the process in Figure 5(b) is to be split, the division could be done by, for example, performing a division onto two sheets. In Figure 5(c), the *center band* is visualized by a red rectangle. Connecting objects that are contained in the cut graph are marked in yellow color. In Figure 5(d) one can inspect the two resulting subgraphs. The replacement objects are inserted and appear as smaller blue objects.

Note, that although all figures use the same scale, the sum of the area of the resulting subgraphs in Fig. 5(d) is significantly smaller than that of the input graph. This positive effect is attributed to the application of the sketch-driven layout algorithm after cutting the graph.

5 Summary

In this paper, we presented our tool "BPMN-Layouter" and its capabilities towards automatic layout for BPMN models.

The requirements BPMN layouts should comply with, were explored in Section 2. In Section 3, we showed that existing tools are not sufficient regarding these requirements. In Section 4, we described the features of our tool and documented the layout techniques in use. Moreover, we introduced the two concepts of assisted insertion of links and divisions of layouts that improve the visualizations of BPMN layouts.

Future work will comprise the integration of the tool and its features into an established designer, e.g. the ORYX-Editor, developed at HPI Potsdam [28]. Additionally, our layout approach is to extend further towards semantics of BPMN, e.g. an intermediate event that is bound to a task/subprocess is a special case when mapping a BPMN-model to a graph. In this case, there is no connecting object (*edge* in graphtheoretic terms) between two flow objects but there exists a tight semantic correlation between the event and the task/subprocess. This requires further work on our approach.

References

[1] OMG, "Business Process Modeling Notation (BPMN) Specification," Object Management Group, January 2009. [Online]. Available: http://www.omg.org/spec/BPMN/1.2/PDF

[2] S. A. White, "Introduction to BPMN," IBM Corporation, 2004. [Online]. Available: http://www.bpmn.org

[3] H. C. Purchase, J.-A. Allder, and D. A. Carrington, "User preference of graph layout aesthetics: A UML study," in *Proc. of the 8th International Symposium on Graph Drawing (GD '00)*, ser. LNCS. Springer-Verlag, 2001, pp. 5–18.

[4] H. C. Purchase, "Which aesthetic has the greatest effect on human understanding," in *Proc. of the 5th Symposium on Graph Drawing (GD '97)*, ser. LNCS, vol. 1353. Springer, 1997, pp. 248–261.

[5] R. Tamassia, G. DiBattista, P. Eades, and I. Tollis, *Graph Drawing*. Prentice Hall, 1999.

[6] M. K. Coleman and D. S. Parker, "Aesthetics-based graph layout for human consumption," *Software – Practice and Experience*, vol. 26, no. 12, pp. 1415–1438, 1996.

[7] M. Kaufmann and D. Wagner, Eds., *Drawing Graphs: Methods and Models*, ser. LNCS Tutorial. Springer, 2001.

[8] S. W. Ambler, *The Elements of UML 2.0 Style*. Cambridge University Press, 2005.

[9] W. Huang, S.-H. Hong, and P. Eades, "Effects of sociogram drawing conventions and edge crossings in social network visualization," *J. Graph Algorithms Appl.*, vol. 11, no. 2, pp. 397–429, 2007.

[10] M. Eiglsperger, C. Gutwenger, M. Kaufmann, J. Kupke, M. Jünger, S. Leipert, K. Klein, P. Mutzel, and M. Siebenhaller, "Automatic layout of UML class diagrams in orthogonal style." *Information Visualization*, vol. 3, no. 3, pp. 189–208, 2004.

[11] M. Siebenhaller and M. Kaufmann, "Drawing activity diagrams," Wilhelm-Schickard-Institut, Tech. Rep. WSI-2006-02, 2006.

[12] H. Eichelberger, "SugiBib," in *Proc. of the 9th International Symposium on Graph Drawing (GD '01)*, ser. LNCS, vol. 2265. Springer, 2002, pp. 467–468.

[13] K. Wittenburg and L. Weitzman, "Qualitative visualization of processes: Attributed graph layout and focusing techniques," in *Proc. of the 4th International Symposium on Graph Drawing (GD '96)*, ser. LNCS, vol. 1190. Springer, 1997, pp. 401–408.

[14] J. M. Six and I. G. Tollis, "Automated visualization of process diagrams," in *Proc. of the 9th International Symposium on Graph Drawing (GD '01)*, ser. LNCS, vol. 2265. Springer, 2002, pp. 45–59.

[15] S. Stein and E. Hagen, "Adopting BPMN with ARIS," 2009, ARIS Expert Paper. [Online]. Available: http://www.ids-scheer.com/set/6473/EBPM%20-%20Stein%20-%20Hagen%20-%20B%PMN%20with%20ARIS%20-%20AEP%20en.pdf

[16] R. Wiese, M. Eiglsperger, and M. Kaufmann, "yfiles: Visualization and automatic layout of graphs," in *Proc. of the 8th Symposium on Graph Drawing (GD '01)*, vol. LNCS 2265. Springer, 2001, pp. 453–454.

[17] M. Siebenhaller, "Orthogonal drawings with constraints: Algorithms and applications," Ph.D. dissertation, Wilhelm-Schickard-Institut, University of Tuebingen, 2009, publication date: summer 2009.

[18] N. Palmer, "Understanding the BPMN-XPDL-BPEL value chain," *Business Integration Journal*, vol. November/Dezember, pp. 54–55, 2006.

[19] U. Brandes, M. Eiglsperger, I. Herman, M. Himsolt, and M. Marshall, "Graphml progress report - structural layer proposal," in *Proc. of the 9th Symposion on Graph Drawing (GD '01)*. Springer Verlag, 2001, pp. 501–512.

[20] K. Sugiyama, S. Tagawa, and M. Toda, "Methods for visual understanding of hierarchical system structures," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. SMC-11,2, pp. 109–125, 1981.

[21] R. Tamassia, "On embedding a graph in the grid with the minimum number of bends," *SIAM Journal on Computing*, vol. 16(3), pp. 421–444, 1987.

[22] P. Effinger, "Automatisches Layout von Geschäftsprozessen," Diplomarbeit, Eberhard-Karls-Universität Tübingen, Wilhelm-Schickard-Insitute, Sand 13, May 2008.

[23] P. Effinger, M. Kaufmann, and M. Siebenhaller, "Enhancing visualizations of business processes," in *Proc. of the 16th International Symposium on Graph Drawing (GD '08)*, ser. LNCS, 2009, pp. 437–438.

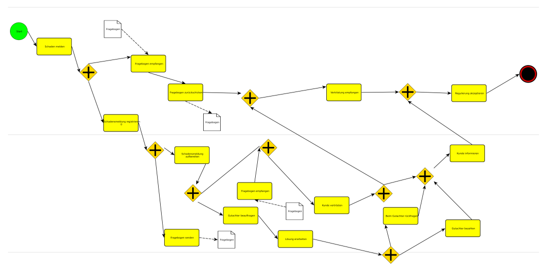
[24] U. Brandes, M. Kaufmann, D. Wagner, and M. Eiglsperger, "Sketch-driven orthogonal graph drawing," in *Proc. of the 10th International Symposium on Graph Drawing (GD '02)*, vol. LNCS 2528. Springer, 2002.

[25] H. Reijers and J. Mendling, "Modularity in process models: Review and effects," in *Business Process Management*, 2008, pp. 20–35. [Online]. Available: http://dx.doi.org/10.1007/978-3-540-85758-7_5

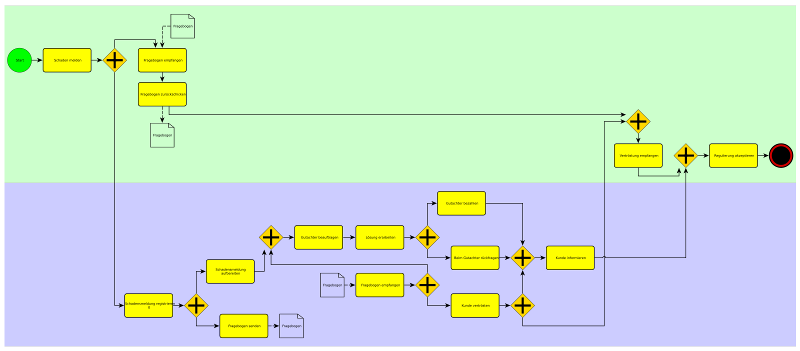
[26] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein, *Introduction to Algorithms, Second Edition*. The MIT Press, September 2001.

[27] P. Effinger, M. Siebenhaller, and M. Kaufmann, "Improving business process visualizations," Wilhelm-Schickard-Institut, Eberhard-Karls-Universität Tübingen, Tech. Rep. 02, 2009. [Online]. Available: http://tobias-lib.ub.uni-tuebingen.de/volltexte/2009/3737

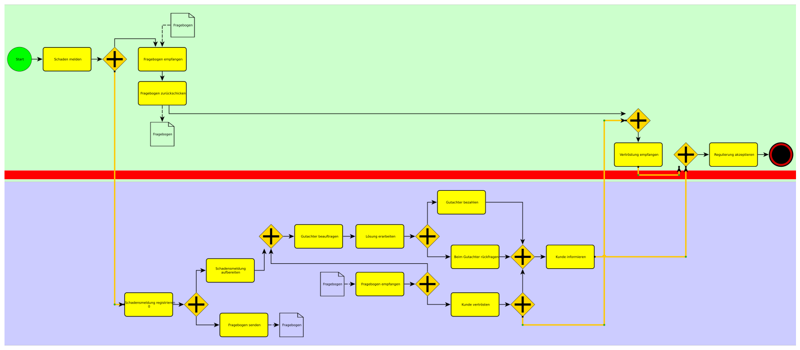
[28] G. Decker, H. Overdick, and M. Weske, "ORYX - an open modeling platform for the BPM community," in *Proc. of the 6th Int'l Conference on Business Process Management (BPM 2008)*, ser. LNCS, M. Dumas, M. Reichert, and M. C. Shan, Eds. Milano, Italy: Springer Verlag, 2008, pp. 382–385.



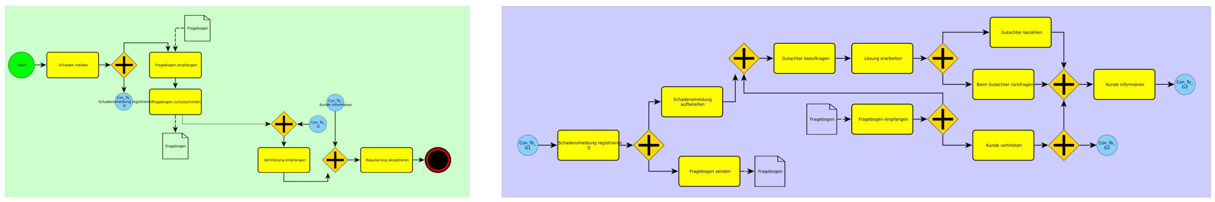
(a) A sketch showing a *BPMN* model with two roles in the process 'notification of claim'.



(b) Computed layout graph.



(c) Layout graph before vertical division; the *center band* is marked red.



(d) Models after division; Again, off-page connector objects (links), depicted in blue, are inserted for division connections.

Figure 5: Example of a BPMN-Layout and a vertical division for a two-role (insurer/insurant) german insurance process showing 'notifications of claims'.