Journal of Visual Languages and Computing 25 (2014) 117–132

Contents lists available at ScienceDirect

Journal of Visual Languages and Computing

journal homepage: www.elsevier.com/locate/jvlc

A linear time layout algorithm for business process models[☆]

Thomas Gschwind[a,*], Jakob Pinggera[b], Stefan Zugal[b], Hajo A. Reijers[c,d], Barbara Weber[b]

[a] IBM Research, Zurich, Switzerland
[b] University of Innsbruck, Austria
[c] Eindhoven University of Technology, The Netherlands
[d] Perceptive Software, The Netherlands

ARTICLE INFO

Article history:
Received 6 September 2012
Received in revised form
5 June 2013
Accepted 14 November 2013
Available online 22 November 2013

Keywords:
Layout
Graph
Workflow
Workflow languages
Business process model

ABSTRACT

The layout of a business process model influences how easily it can be understood. Existing layout features in process modeling tools often rely on graph representations, but do not take the specific properties of business process models into account. In this paper, we propose an algorithm that is based on a set of constraints which are specifically identified toward establishing a readable layout of a process model. Our algorithm exploits the structure of the process model and allows the computation of the final layout in linear time. We explain the algorithm, show its detailed run-time complexity, compare it to existing algorithms, and demonstrate in an empirical evaluation the acceptance of the layout generated by the algorithm. The data suggests that the proposed algorithm is well perceived by moderately experienced process modelers, both in terms of its usefulness as well as its ease of use.

1. Introduction

Business process models serve a wide variety of purposes where a distinction can be made between models that are to be read by *humans* versus those to be read by *machines* [6]. In the latter case, one may think of *workflow specifications*, as enacted by process-aware information systems, or *simulation models*, which are used to estimate a certain measure's effect. Our concern in this paper, however, is with the former category, i.e., those models that are studied by humans to make sense of how organizational operations are related to one another. It is crucial that such models are understandable to end users from a variety of backgrounds [6].

The *graphical layout* of a process model has been named as affecting the ease with which a human reader can access the information in such a model [26,29]. When one is concerned with simplifying the task of reading a process model, layout seems a highly attractive angle. After all, in many situations, the graphical positioning of model elements is at the discretion of the modeler, who is creating the model, or even the user, who wants to read the model. Also, layout, as part of what is known as a model's *secondary notation* [22], does not affect the formal meaning of the model. In this way, focusing on a model's layout allows for a separation of concerns with respect to other quality aspects (e.g., a model's semantic quality).

A considerable body of knowledge exists on how to layout graphical models in order to improve their readability [24,25]. What is currently missing is a clear understanding of how the specific characteristics of business process models can be taken into account when applying these insights. This hampers the transfer of such knowledge to its practical application. In that respect, it is telling that despite a huge availability of advanced process modeling tools (e.g., IBM Blueworks, Oryx, BPMOne), none of these provide automated layout features beyond elementary alignment operations.

☆ This paper has been recommended for acceptance by S.-K. Chang.
* Corresponding author. Tel.: +41 44 724 8990; fax: +41 44 724 8953.
E-mail address: thg@zurich.ibm.com (T. Gschwind).

1045-926X/$ - see front matter © 2013 Elsevier Ltd. All rights reserved.
http://dx.doi.org/10.1016/j.jvlc.2013.11.002

To pick up on this demand, this paper presents an efficient algorithm for clearly laying out business process models. Our approach has been to identify a set of favorable layout constraints from literature, which we subsequently used as the basis for the development of an algorithm that enforces these constraints. The process structure tree as presented in [32] plays an important role in this algorithm, as it provides the hierarchical abstraction of the semantics of the underlying business process. Furthermore, we have validated our theoretical results both in terms of measuring actual performance, the extent to which the proposed algorithm is perceived as useful, and we conducted a comparison between the proposed algorithm with existing ones.

The contribution of the proposed algorithm is to ensure the application of state-of-the-art knowledge to display business process models in a clear and concise way. The approach has been developed for BPMN models that are modeled from left-to-right, but the presented insights can be easily transferred to other flow-oriented languages as well as other model orientations. Our vision is that the algorithm is picked up to be implemented in various electronic modeling environments, in this way improving the quality of the models being created *without* putting an extra load on the modelers themselves. In fact, the provided support may even alleviate a modeler's task, particularly in situations when her modeling experience is limited. When process models are being used within an electronic environment, which is an increasingly realistic option [20], the proposed algorithm directly supports the reader herself in arranging the model in a highly readable form (regardless of the original layout).

The remainder of this paper is structured as follows. Section 2 introduces basic concepts used throughout this paper. Section 3 then elaborates on the layout constraints that seem sensible to follow when laying out business process models. Section 4 introduces the algorithm that takes these constraints into account to provide automated layout support and discusses the algorithm's complexity. Subsequently, Section 5 reports our evaluation of the proposed algorithm in terms of actual run-time performance, its perceived ease of use and usefulness, and gives a comparison of the proposed algorithm with existing ones. Section 6 examines related work. Finally, Section 7 concludes the paper with a summary and outlook.

2. Background

Process models typically describe in a graphical way the activities, events, states, and control flow logic that constitute a business process [3]. Additionally, process models may also include information regarding the involved data, organizational and IT resources, and even other artifacts such as external stakeholders and performance metrics, see e.g. [28].

The process model depicted in Fig. 1, for example, consists of a *start event*, and an *end event*, four *activities* (i.e., a, b, c, and d), four *gateways*, and a set of *control-flow edges* connecting these elements. Gateways can be either splitting nodes (i.e., nodes with one incoming and multiple outgoing arcs, like gateways g and j) or joining gateways (i.e., nodes with multiple incoming and one outgoing arc, like gateways h and i).

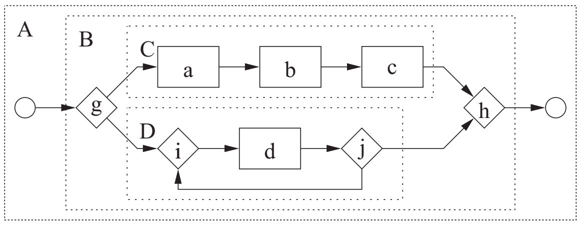

Fig. 1. A process model and its major sub-fragments.

Similar to other forms of conceptual modeling, process models are often required to be intuitive and easily understandable, especially in the phases of information systems projects that are concerned with requirements documentation and communication [6,10].

The graph layouting algorithm proposed in this paper is based on the idea of decomposing a process model into single-entry single-exit (SESE) fragments. Fig. 1 shows an exemplary SESE decomposition, which we will use throughout this paper. The decomposition of this model results in SESE fragments A, B, C and D. As the name suggests, each SESE fragment has *exactly one* incoming and *exactly one* outgoing edge, irrespective of the internal structure of the fragment. For instance, the internal structure of fragment C is a sequence of activities, whereas fragment D consists of a branched construct. Furthermore, SESE fragments can be embedded in other SESE fragments: note how fragment B aggregates fragment C and fragment D to a SESE fragment.

SESE fragments can be characterized as structured or unstructured. Structured fragments are composed of *blocks*, which may be nested, but must not overlap; i.e., their nesting must be regular [32]. Thereby, a *block* refers to a SESE fragment. In general, structured fragments can be classified as *sequences*, *branching fragments*, *atomic*, and *structured loops. Sequences* consist of a sequence of activities or fragments (cf. fragments A and C in Fig. 1). *Branching fragments* consist of a diverging (i.e., splitting) gateway as entry and a converging (i.e., joining) gateway as exit. To illustrate a branching fragment, we refer to fragment B in Fig. 1. *Atomic* fragments cannot be further subdivided into fragments and represent gateways and individual nodes. *Structured loops* consist of a converging gateway followed by an optional fragment, in turn, followed by a diverging gateway. The latter has an exit branch and one or more branches that loop back (cf. fragment D in Fig. 1).

Within unstructured fragments, on the other hand, not all blocks are regularly nested. Depending on whether or not they can be laid out without edge-crossings, they are denoted as either *planar* or *non-planar*.

3. Constraints for laying out process models

An important consideration when laying out business process models is the set of constraints that should be followed. After a review of the literature, we have identified constraints for laying out business process models [19,24,29], as well as trees [30]. Based on this survey, we