A Visualization Concept for High-Level Comparison of Process Model Versions

Simone Kriglstein[1] and Stefanie Rinderle-Ma[2]

[1] SBA Research, Vienna, Austria
SKriglstein@sba-research.at

[2] University of Vienna, Faculty of Computer Science,
Research Group Workflow Systems and Technology, Vienna, Austria
stefanie.rinderle-ma@univie.ac.at

Abstract. Managing large collections of different process model versions is for many organizations inevitable and results from, e.g., adaption of models to solve different challenges or modifications due to changed or new conditions. One challenge in this context is to make the differences between the versions visible and comparable. Visualizations have the advantage that they can present the relationships between the different process versions in a user-friendly way and therefore support users in their decisions. In this paper we introduce a visualization concept with the goal to provide a simple overview in order to compare complementary or contrasting characteristics between different versions. The design idea is presented on the basis of two use cases. Limitations of the concept are also discussed. The visualization concept should support users to gain a first impression about the characteristics between the versions and can be used as an entry point for a more detailed analysis of the different versions.

Key words: Visualization, Version management, Business process model collection

1 Introduction

In the last years, different approaches have been developed to support organizations in adjusting their business processes in order to react flexible on changing environmental conditions (e.g., new or changed requirements) or on unplanned events/exceptions (e.g., to correct design errors) [1]. It is often unavoidable that organizations have large collections of large numbers of different business process versions. Therefore, it is common that organizations have to maintain repositories that can contain hundreds or even thousands of versions usually created by different users with diverse goals and responsibilities [2]. For graphical process notations, process evolution is mainly carried out by applying change operations (e.g., adding, deleting, or moving a process fragment) that are defined over the graph structure of the process model. However, in many cases it is not only sufficient to make the applied change operations comparable between the different versions, but also to consider security-relevant information such as who conducted changes or who has permissions to apply change operations. Such kind of information plays an important role, especially for collaborative modeling (e.g., to control inadvertent changes in process models by users who are not designated as editors).

Over the years, approaches (e.g., [2, 3, 4, 5]) have been developed to make the management of versions easier in order to reduce costs and efforts. For example, version management supports companies to re-use processes as reference models and adapt them to their requirements, e.g., by applying pre-defined change operations on the reference process [6]. Further reasons for their increasing popularity are that most users are familiar with version management concepts (e.g., to manage changes of documents, computer programs, and wikis) and that they provide features (e.g., to roll back to any versions at any point in time) to recover process models from mistakes and to see additional security-relevant information (e.g., who changed what and when).

Among other things, visualizations support users to better understand the datasets and to recognize patterns (e.g., detection of data groups) which can be helpful for their further decisions [7]. The potential of visualizations to make things (e.g., patterns, relationships, or anomalies) visible makes them attractive as an additional support for the analysis and management of versions. For example, a graphical representation can help to see which change operations were applied between versions and to compare them with the number of users who conducted these changes. Or a visualization can be used to make the number of users who conducted changes in regard to the assigned permission operations visible.

This paper introduces a visualization concept in order to compare complementary or contrasting characteristics between different process versions. The concept is inspired by Shneiderman's Visual Information-Seeking Mantra *"Overview first, zoom and filter, then details-on-demand*" [8] that describes how data should be presented to make it most effective for users. Based on the Visual Information-Seeking Mantra, the design idea is to develop an approach that provides a simple overview which can be used as starting point in order to move from summary information to detailed information. Furthermore, the ability of human's perception for detection of patterns is taken into account in order to gain a first impression of the visualized characteristics between the different versions.

The remainder of the paper is structured as follows. In Section 2 related work is discussed and Section 3 gives a short overview about the relevant aspects of graphical perception. The visualization concept is introduced in Section 4 and in Section 5 two use cases are presented. Limitations of the visualization concept and future work are discussed in Section 6. Finally, the paper is concluded in Section 7.

2 Related Work

Although version-based visualization approaches are well-known in several fields of computer science (see e.g., [9, 10, 11, 12, 13, 14, 15, 16, 17, 18]), the development of visualization approaches for version management in the business process context has received little attention in the last years [19, 20]. Usually the different versions are presented as a list (e.g., Apromore platform [21] or IBM Business Process Manager [22]), as an indented list to highlight the hierarchical structure (e.g., e*Insight Business Process Manager [23]) or as a version graph to visualize the hierarchical and non-hierarchical relationships between versions (e.g., [24, 25, 26]).

For a comparative analysis of different versions, multiple views are often used (see e.g., [22, 23, 25, 26]). For example, one view gives an overview about the stored versions (e.g., as a list or version graph) and the detail information of the selected version (e.g., change information and/or corresponding process model) is presented in another view. The conducted literature review and survey in [27] shows that change information is usually presented in tables or directly highlighted in the process graph via visual properties, like colors. Although the usage of multiple views allows to simplify the design, users need time to orientate themselves after switching between different views in order to comprehend the context and to compare the different views [28, 29]. The focus of our approach is to provide a first overview about version information in order to support a simple comparative analysis among the different versions. Furthermore, it can be integrated into approaches such as the Apromore platform. [4].

3 Background Knowledge: Graphical Perception

In this section, we shortly present relevant aspects of graphical perception which are from interest for our approach. The design considers the ability of human's perception to recognize changes, e.g., in color and length in order to detect simple patterns in categorical datasets. Studies (e.g., [30, 31]) show that pre-attentive processing can guide the human's attention toward target objects by visual properties at a glance without effort [32, 33, 34]. Hence, objects seem to pop out from the rest of the scene provided that the target object is distinguished from the other objects by simple features (e.g., colors) [33, 34, 35]. For example, it does not take any effort to see the black objects in Figure 1. Furthermore, the example shows well that the black objects can be split into two groups because of their spatial proximity which is one of the Gestalt laws. The Gestalt laws (e.g., proximity, similarity, closure, and symmetry) describe rules how the space has to be organized to see patterns in visual displays [36]. However, the number of visual properties that can be used is limited and pre-attentive processing is not for all tasks sufficient [7]. Typically, pre-attentive processing is combined with attentive processing for the analysis of patterns in order to actively scan serially through the display to find the target object [31, 34, 35]. For example, to find the target object in Figure 1 (depicted in the thought bubble) the attention

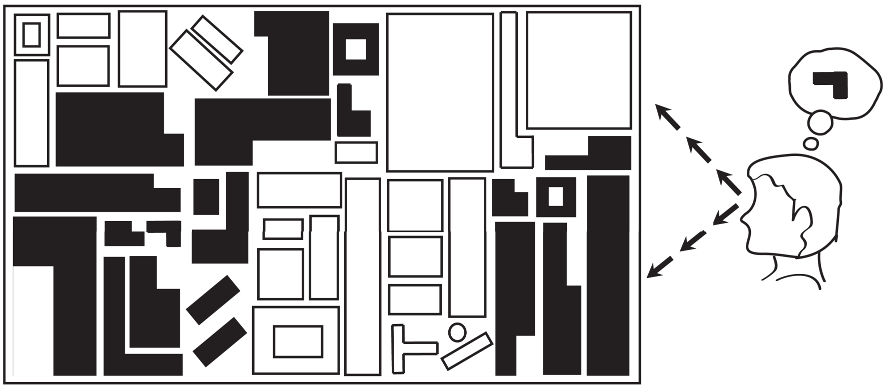

Fig. 1. Example for pre-attentive and attentive processes: The attention is guided to locations that have the same features (e.g., color black) as the target object (shown in the thought bubble) in order to scan these areas serially.

is first guided to areas which have the same features (e.g., color, shape, and orientation).

4 Design Approach

In this section we present the concept of our visualization approach. The idea is to provide an overview of categorical datasets to help users to compare parts of the data in order to support simple comparative analysis and to provide an entry point that allows users to navigate to areas of interest.

4.1 Basic Concept

The main components are stacked bar charts which are similar to bar charts. Bar charts can be used for various datasets and are one of the most common chart types [37]. In contrast to bar charts, stacked bar charts allow the comparison of numeric values in regard to their corresponding subcategories. Figure 2 shows the concept of our approach. The version numbers are presented chronologically on the y-axis and the subcategories are presented as horizontal bars. The juxtaposition of two stacked bar charts makes it possible to compare complementary or contrasting characteristics between different process versions.

4.2 Visual Features

The length of a single bar presents the value of the corresponding subcategory. The length of the total stacked bar reflects the sum of the subcategories. Different colors are used to indicate different subcategories. Based on Ware's recommendation [33] it is necessary to consider:

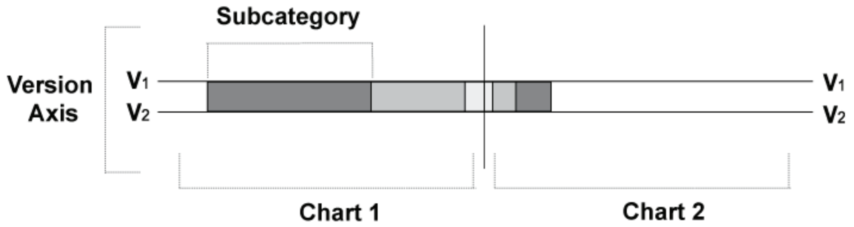

Fig. 2. Basic structure of the concept.

- *Visual Distinctness*: The stacked bars and their subcategories should pop out from the scene to support effective visual search and comparison. In other words the contrast between the background and foreground colors should be as high as possible.
- *Learnability*: Colors have to be used consistently in order to be clear that one color presents a specific subcategory.

The number of colors is restricted in order to avoid effects like eyestrain, confusion or disorientation [33, 38]. There exist several recommendations (e.g., [33, 38]) about how many colors can be used. The recommendations vary from case to case, but often the maximum number is between 6 and 12 colors [33]. Furthermore, it is recommended to use colors that are widely spaced along the color spectrum [39]. For example, in addition to black and white, the following colors are recommended [33, 39]: red, yellow, green, blue, brown, pink, orange, grey, and purple.

The different lengths of bars and colors support users to gain a first impression about the distribution of the subcategories and allow users to detect simple patterns between the different versions. For example, to see which versions have high or low differentiation in regard to a specific subcategory or if the values of a specific subcategory increase/decrease over the versions. Furthermore, it can be used to monitor activities between versions and it helps to detect irregularities, unexpected activities or activities which are not allowed (e.g., if more users made changes than users who had the edit permission). Figure 3 shows an example for the interplay between lengths of bars and colors. The colors blue, red, and green are used to present three different subcategories. The different lengths of bars show, e.g., that the subcategory 3 (color green) has the highest value between version 1 (V1) and version 2 (V2) in chart 1. Similarly, subcategory 1 (color blue) has the highest value between version 2 (V2) and version 3 (V3) in chart 2. Moreover, the comparison of chart 1 and chart 2 shows that the value of subcategory 2 (color red) has the biggest differentiation between version 1 (V1) and version 2 (V2).

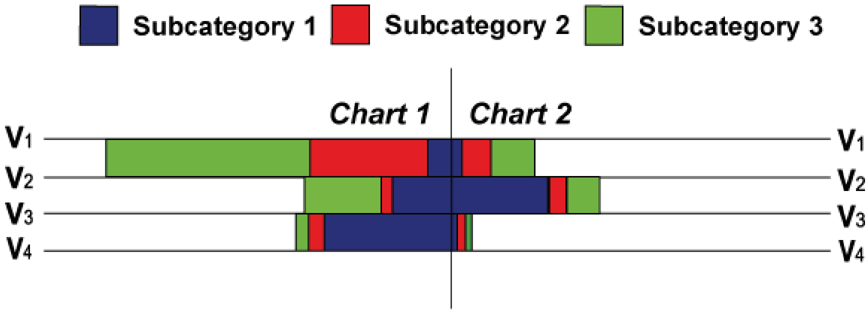

Fig. 3. The example presents the juxtaposition of chart 1 with chart 2 for the analysis of the differences between version 1 (V1), version 2 (V2), version 3 (V3), and version 4 (V4) in regard to three subcategories that are color-coded with blue, red, and green.

5 Use Cases

In this section we present two use cases with the aim to show possible applications as an aid to understand the above described design concept better. In the first use case, we juxtapose the change operations with the number of users who conducted the changes. The focus of the second use case is on the number of permission assignments and the number of users who conducted a permission operation between two versions.

5.1 Use Case 1: Change Operations

The results of one of our previous studies [27] confirmed that the comparison between versions in regard to change operations is often from interest. However, it was also mentioned as helpful to see additional information, like, the users who conducted the changes. Such information can be especially from interest for collaboration modeling and shared processes.

Figure 4 shows an example how both information can be presented. The choice of change operations is based on the following six change patterns that are defined by Weber et al. [40]: *Insert*, *Delete*, *Move*, *Replace*, *Swap*, and *Copy*. These change operations are defined as subcategories. The left chart in Figure 4 presents the distribution of change operations and the right chart shows the distribution of the number of users who performed the change operations between versions. The values of the bars for the corresponding change operations present the number of conducted change operations between the versions (c.f. left chart in Figure 4) and the distinct number of users (c.f. right chart). The representation supports users to answer simple questions such as, which change operations were conducted, which changes were made more/less frequently between which versions, how many users made changes between which versions, or which change operations were executed by more/less users. The juxtaposition of the conducted change operations with the number of users allows to see

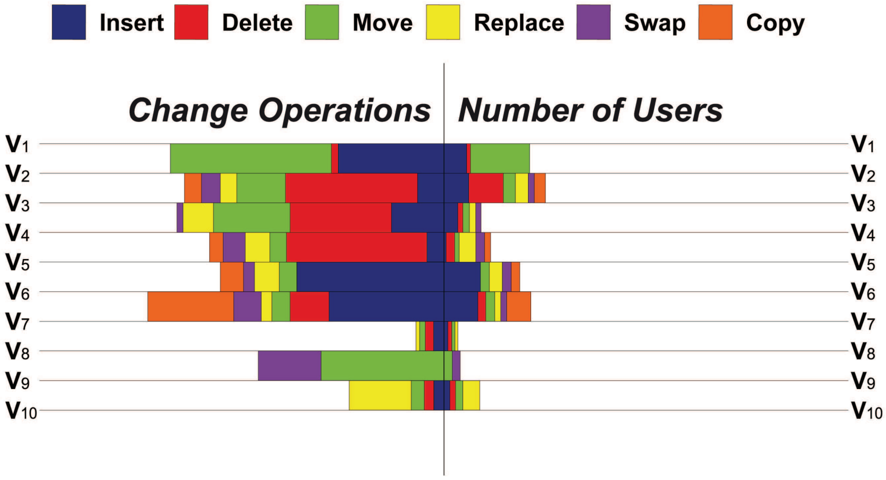

Fig. 4. Example for use case 1: Juxtaposition of change operations with the number of users who conducted the changes between the versions (V1-V10).

them not only separately but also in combination. For example, although the sum of change operations are approximately equal between the first stacked bar and the third stacked bar, more users made changes between V1 and V2 than between V3 and V4. Furthermore, each stacked bar between two versions can be analyzed individually, e.g., the change operations *Insert*, *Delete*, and *Move* were conducted between V1 and V2 and between V8 and V9 the change operations *Move* and *Swap* were made. Moreover, the analysis of the overall picture shows, e.g., that the change operation *Delete* builds a cluster between V2 and V5. Such kind of information can introduce further and deeper questions – e.g., Why did more users made changes? or Why were many *Delete* operations conducted? – that can be answered in a detailed view (e.g., after a stacked bar is selected, the process with change information can be presented in another view).

5.2 Use Case 2: Permission Assignments

Over the years, controlling and monitoring of permission assignments become more and more important, especially for shared processes. One research focus is on the comparison between users' activities and permission assignments in order to detect possible discrepancies [41].

The presented approach in Section 4 can be used to provide an overview about the number of permission assignments and the actual number of users who conducted the permission operations between the versions. Figure 5 presents an example with the permission operations as subcategories. The permission operations define users' authorizations and for the example the suggested operations by Leitner et al. [42] are used: *Add*, *Delete*, *Execute*, and *Monitor*. The left

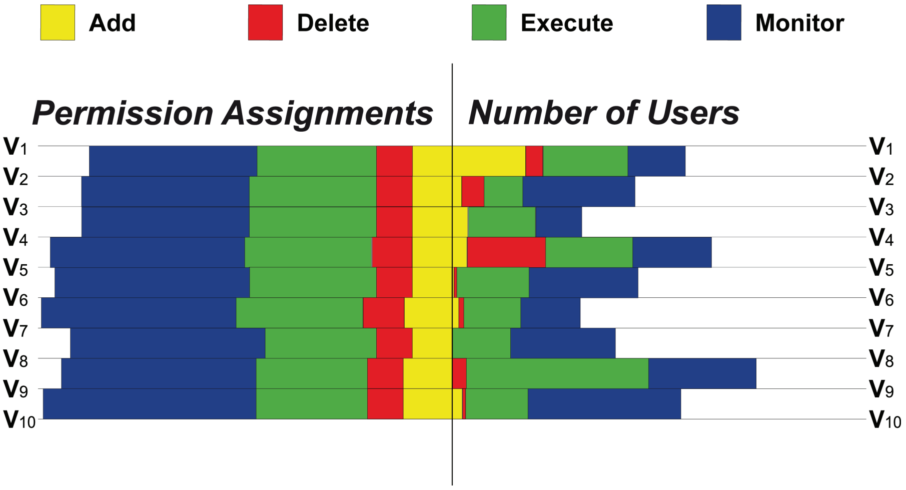

Fig. 5. Example for use case 2: Juxtaposition of the number of permission assignments with the number of users who conducted permission operations between the versions (V1-V10).

chart shows the number of assignments of permission operations to users and the right chart presents the distinct number of users who conducted operations between the versions. The visualization shows that only little differentiations of permission assignments to their users exist between the versions (see left chart in Figure 5). Therefore, changes of permission assignments between versions (e.g.. between V6 and V8 in Figure 5) stand out from the relative stable representation. The juxtaposition of the number of permissions assignments with the number of users who conducted permission operations allows to see possible discrepancies between the versions. For example, in Figure 5 more users conducted the *Add* operation between V1 and V2 than the number of authorized users who are assigned to this operation. Further possible discrepancies can be found between V4 and V5 for the *Delete* operation as well as between V8 and V9 for the *Execute* operation. Although, further information is necessary for a detailed analysis (e.g., to find out the reasons for these discrepancies), such an overview allows a fast detection of interesting areas. However, detection of discrepancies is only possible if the number of users who conducted permission operations is higher than the number of permission assignments. Therefore, the usage of additional visual hints can be helpful in case a conflict exist between the versions (see Figure 6 for an example). Further limitations are discussed in the next section.

6 Limitations and Future Work

In this section we discuss limitations of the presented design concept and our next steps.

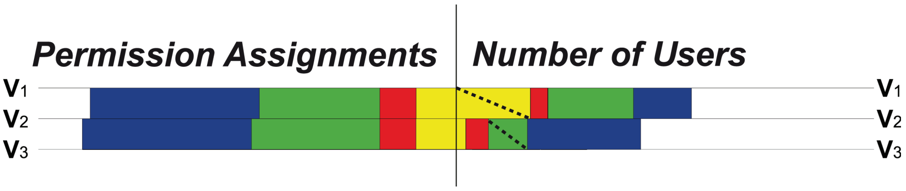

Fig. 6. To highlight subcategories with discrepancies, a dotted line is used as additional visual hint.

One limitation is that the linear representation of versions in chronological order only allows to analyze versions which are connected directly. For example, a comparison between V2 and V8 in Figure 4 is not possible. Furthermore, it is not possible to compare different branches from one version. Hence it is necessary not only to provide a static visualization but also interactivity. One of our next steps is to find ways to integrate interactivity in our concept, e.g., to select the versions for comparison and the aggregated information is presented in a shared stacked bar.

A further limitation is that only the juxtaposition of two characteristics is possible. For the analysis of more than two characteristics, small multiples [43] can be used as series of paired stacked bar charts to present different juxtapositions of two characteristics between versions. The advantage of small multiples is that multidimensional data can be visualized without packing all information into a single chart.

According to Ware [33, 36] the skill to detect complex patterns in visualizations has to be learned by the viewers. Therefore, it is necessary that the visualization is designed in such a way that supports users to easily identify patterns. Ware [36] recommends to consider pattern-finding skills (e.g., based on graphical perceptual capabilities) that are common for the viewers. Therefore, we plan to evaluate our design concept in order to investigate how well viewers can identify patterns and to detect possible improvements.

An increasing number of subcategories and therefore an increasing number of bars makes the comparison – especially for bars which are further away from the midline (y-axis) – more difficult. Hence, we will investigate if the number of bars influence the identification of patterns. Moreover, we want to find out the maximal number of bars which can be used.

The use cases in Section 5 show that visual hints can be helpful to visualize additional information (e.g., to make discrepancies clearly visible). However such additional visual hints can influence the detection of patterns. Therefore we plan to find solutions how we can use such visual hints in order to reduce negative effects on the identification of patterns.

7 Conclusion

In this paper we suggested a visualization concept for a high-level comparison of complementary or contrasting characteristics between different process model versions. The concept based on the Shneiderman's Visual Information-Seeking Mantra to provide an overview in order to detect areas of interest that can be used as starting point for further detailed analysis of the process versions (e.g., with the help of detail-on-demand techniques). Furthermore, the presented approach can be used as add-on feature to the commonly used lists that present the different versions in repositories. This supports users to answer simply questions, like, between which versions more or less change operations were made. We introduced the concept on the basis of two use cases to illustrate the diversity of situations where the approach can be useful. However, the concept also shows several limitations (e.g., only representation of chronological order or limitation of subcategories) that need further investigations. Moreover, we also plan to evaluate the visualization concept with users to improve the concept further.

References

1. Rinderle, S., Reichert, M., Dadam, P.: Correctness criteria for dynamic changes in workflow systems: A survey. Data and Knowledge Engineering **50**(1) (2004) 9–34
2. Ekanayake, C.C., Rosa, M.L., Ter Hofstede, A.H.M., Fauvet, M.C.: Fragment-based version management for repositories of business process models. In: Proc. of the 2011th Confederated international Conference on On The Move to Meaningful Internet Systems - Volume Part I (OTM'11), Springer (2011) 20–37
3. Fauvet, M.C., Rosa, M.L., Sadegh, M., Alshareef, A., Dijkman, R.M., Garcia-Banuelos, L., Reijers, H.A., van der Aalst, W.M., Dumas, M., Mendling, J.: Managing process model collections with AProMoRe. In: Proc of the 8th International Conference on Service Oriented Computing (ICSOC 2010), Springer (2010)
4. Rosa, M.L., Reijers, H.A., van der Aalst, W.M., Dijkman, R.M., Mendling, J., Dumas, M., García-Bañuelos, L.: APROMORE: an advanced process model repository. Expert Systems with Applications **38**(6) (2011) 7029–7040
5. Yan, Z., Dijkman, R., Grefen, P.: Business process model repositories - framework and survey. Inf. Softw. Technol. **54**(4) (2012) 380–395
6. Hallerbach, A., Bauer, T., Reichert, M.: Capturing variability in business process models: the Provop approach. Journal of Software Maintenance and Evolution: Research and Practice **22**(6-7) (2010) 519–546
7. Fekete, J.D., Wijk, J.J., Stasko, J.T., North, C.: The value of information visualization. In Kerren, A., Stasko, J.T., Fekete, J.D., North, C., eds.: Information Visualization. Springer (2008) 1–18
8. Shneiderman, B.: The eyes have it: a task by data type taxonomy for information visualizations. In: Proc. of the 1996 IEEE Symposium on Visual Languages, IEEE Computer Society (1996)
9. Collberg, C., Kobourov, S., Nagra, J., Pitts, J., Wampler, K.: A system for graph-based visualization of the evolution of software. In: Proc. of the 2003 ACM Symposium on Software Visualization, ACM Press (2003) 77–86
10. Diehl, S.: Visualizing the evolution of software systems. In: Software Visualization. Springer (2007) 129–147

11. Eick, S.G., Graves, T.L., Karr, A.F., Mockus, A., Schuster, P.: Visualizing software changes. IEEE Trans. Softw. Eng. **28** (2002) 396–412
12. Eick, S.G., Steffen, J.L., Sumner, Jr., E.E.: Seesoft - a tool for visualizing line oriented software statistics. IEEE Trans. Softw. Eng. **18** (1992) 957–968
13. Gracanin, D., Matkovic, K., Eltoweissy, M.: Software visualization. Innovations in Systems and Software Engineering (ISSE) **1**(2) (2005) 221–230
14. Langelier, G., Sahraoui, H., Poulin, P.: Exploring the evolution of software quality with animated visualization. In: Proc. of the IEEE Symposium on Visual Languages and Human-Centric Computing, IEEE Computer Society (2008) 13–20
15. Maletic, J.I., Marcus, A., Feng, L.: Source viewer 3D (sv3D): a framework for software visualization. In: Proc. of the 25th International Conference on Software Engineering, IEEE Computer Society (2003) 812–813
16. Ogawa, M., Ma, K.L.: Software evolution storylines. In: Proc. of the 5th International Symposium on Software Visualization, ACM Press (2010) 35–42
17. Tu, Q., Godfrey, M.: An integrated approach for studying software architectural evolution. In: Proc. of the International Workshop on Program Comprehension. (2002)
18. Voinea, L., Telea, A., Chaudron, M.R.V.: Version-centric visualization of code evolution. In Proc. of the Eurographics/IEEE-VGTC Symposium on Visualization, Eurographics Association (2005) 223–230
19. Chaâbane, M.A., Andonoff, E., Bouzguenda, L., Bouaziz, R.: Dealing with business process evolution using versions. In: Proc. of the International Conference on e-Business, ICE-B is part of ICETE - The International Joint Conference on e-Business and Telecommunications, INSTICC Press (2008) 267–278
20. Zhao, X., Liu, C.: Version management in the business process change context. In Alonso, G., Dadam, P., Rosemann, M., eds.: Business Process Management. Volume 4714 of LNCS., Springer (2007) 198–213
21. The Apromore Foundation: Apromore platform: v0.2 01/08/2011 (2009–2012) `http://brahms0.imag.fr:8080/Apromore-portal/` <accessed 19.04.2012 >.
22. IBM: IBM Business Process Manager `http://www-01.ibm.com/software/integration/business-process-manager/` <accessed 19.04.2012 >.
23. SeeBeyond Technology Corporation: SeeBeyond eBusiness Integration Suite Primer, Release 4.5.2 (1999–2002) `http://docs.oracle.com/cd/E18867_01/4.5.x/4.5.2/Business_Integration_Suite/Primer.pdf` <accessed 19.04.2012 >.
24. Bae, H., Hu, W., Yoo, W.S., Kwak, B.K., Kim, Y., Park, Y.T.: Document configuration control processes captured in a workflow. Computers in Industry **53**(2) (2004) 117 – 131
25. Cui, D., Feng, G.: Component based design data version management and visualization in IMS-DATE. In: Proc. of the IEEE International Conference on Industrial Engineering and Engineering Management, IEEE Computer Society (2008) 1529–1533
26. Thomas, O.: Version management for reference models: Design and implementation. In Becker, J., Delfmann, P., eds.: Reference Modeling: Efficient Information Systems Design Through Reuse of Information Models. Physica-Verlag (2007) 1–26
27. Kriglstein, S., Rinderle-Ma, S.: Change visualization in business processes - requirements analysis. In: Proc. of the International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications (GRAPP/IVAPP), SciTePress (2012)
28. Betz, S., Eichhorn, D., Hickl, S., Klink, S., Koschmider, A., Li, Y., Oberweis, A., Trunko, R.: 3D representation of business process models. In: Proc. of Modellierung betrieblicher Informationssysteme (MobIS). LNI, GI (2008) 73–87

29. North, C.: Information visualization. In Salvendy, G., ed.: Handbook of Human Factors and Ergonomics, John Wiley & Sons (2005) 1222–1246
30. Treisman, A.: Features and objects in visual processing. Scientific American **255**(5) (1986) 114–125
31. Treisman, A.M., Gelade, G.: A feature-integration theory of attention. Cognitive Psychology **12**(1) (1980) 97–136
32. Fekete, J.D., Plaisant, C.: Interactive information visualization of a million items. In: Proc. of the IEEE Symposium on Information Visualization (InfoVis'02), IEEE Computer Society (2002) 117–124
33. Ware, C.: Visual Thinking: for Design. Morgan Kaufmann Publishers Inc. (2008)
34. Williams, E.: Visual search: a novel psychophysics for preattentive vision. Master's thesis, Research Science Institute (1999)
35. Healey, C.G., Booth, K.S., Enns, J.T.: Visualizing real-time multivariate data using preattentive processing. ACM Trans. Model. Comput. Simul. **5**(3) (1995) 190–221
36. Ware, C.: Information Visualization: Perception for Design. Morgan Kaufmann Publishers Inc. (2004)
37. Yau, N.: Visualize This: The FlowingData Guide to Design, Visualization, and Statistics. John Wiley & Sons (2011)
38. Stone, D., Jarrett, C., Woodroffe, M., Minocha, S.: User Interface Design and Evaluation. Morgan Kaufmann (2005)
39. Galitz, W.O.: The Essential Guide to User Interface Design: Introduction to GUI Design Principles and Techniques. Wiley & Sons (2002)
40. Weber, B., Reichert, M., Rinderle-Ma, S.: Change patterns and change support features - enhancing flexibility in process-aware information systems. Data Knowl. Eng. **66**(3) (2008) 438–466
41. Rinderle-Ma, S., Wil, M.: Life-cycle support for staff assignment rules in process-aware information systems. Technical report (2007)
42. Leitner, M., Rinderle-Ma, S., Mangler, J.: AW-RBAC: Access control in adaptive workflow systems. In: Proc. of the 6th International Conference on Availability, Reliability and Security, IEEE Computer Society (2011) 25–37
43. Tufte, E.R.: Envisioning Information. Graphics Press (1990)