Enabling Personalized Process Schedules with Time-aware Process Views

Andreas Lanz, Jens Kolb, and Manfred Reichert

Institute of Databases and Information Systems, Ulm University, Germany
{andreas.lanz,jens.kolb,manfred.reichert}@uni-ulm.de

Summary. Companies increasingly adopt process-aware information systems (PAISs) to model, enact, monitor, and evolve their business processes. Although the proper handling of temporal constraints (e.g., deadlines and minimum time lags between activities) is crucial for many application domains, existing PAISs vary significantly regarding the support of the temporal perspective of a business process. In previous work, we introduced characteristic time patterns for specifying the temporal perspective of PAISs. However, time-aware process schemas might be complex and hard to understand for end-users. To enable their proper visualization, therefore, this paper introduces an approach for transforming time-aware process schemas into enhanced Gantt charts. Based on this, a method for creating personalized process schedules using process views is suggested. Overall, the presented approach enables users to easily understand and monitor time-aware processes in PAISs.

Key words: Human-centric PAIS, Temporal Perspective, Time Patterns

1 Introduction

Companies strive for improved life cycle support of their business processes [1]. In particular, IT support for modeling, enacting, and analyzing these processes results in competitive advantages [2, 3]. In this context, process-aware information systems (PAISs) offer promising perspectives for process automation. In particular, they allow defining a business process in terms of an explicit *process schema* and executing respective *process instances* in a controlled and efficient manner [1].

Existing PAISs vary significantly regarding their support of the temporal perspective of processes [4, 5]. However, integrating the temporal perspective in a PAIS is crucial, since most business processes must obey temporal constraints [6]. Moreover, in many application domains the proper handling of temporal constraints is vital in order to successfully complete a process (e.g., flight planning, patient treatment, and automotive engineering) [7, 8]. By contrast, different kinds of planning tools (e.g., project management tools) exist, providing sophisticated facilities for handling and visualizing time constraints. However, these tools lack an operational process support, which is needed for proper visualization and systematic analysis of the temporal perspective in PAISs.

Generally, when incorporating the temporal perspective complex and large process schemas may result, which are difficult to comprehend for users. In

particular, it is important that all users involved in a business process are aware of respective temporal constraints [9]. To tackle this challenge, we introduce an approach transforming a time-aware process schema to an extended version of Gantt charts. Note that Gantt charts are well known in project management [10]. In particular, Gantt charts allow users to easily perceive and assess temporal properties of time-aware processes. However, despite their simplicity, representing an entire time-aware process schema as Gantt chart might be complex and inappropriate for end-users. To address this issue, we provide mechanisms for abstracting a process schema (and the respective Gantt chart) to meet specific requirements of a particular user. This allows providing personalized process schedules for each user and thus reducing the complexity of time-aware process schemas.

Sect. 2 discusses related work. Sect. 3 introduces fundamentals required to understand this paper. Sect. 4 describes the transformation of time-aware processes to Gantt charts. Sect. 5 shows how personalized process schedules can be created based on respective Gantt charts. Finally, Sect. 6 concludes the paper.

2 Related Work

Gantt charts have been used for visualizing project plans for a long time [11, 10]. Nowadays, almost all project management tools offer a Gantt chart-based appearance of project plans [10]. Furthermore, extensions to Gantt charts have been proposed in different application domains. Such an approach is provided by AsbruView [12], which was originally introduced for therapy planing; in particular, it extends Gantt charts with minimum/maximum durations and provides basic support for loops as well as conditional routing. Still, it does not consider some of the specific requirements found in the context of PAISs (e.g., time lags between activities).

In turn, considerable research has been conducted concerning the modeling and verification of time-aware processes [13, 5, 14, 15]. However, respective approaches either rely on traditional process appearances (e.g., BPMN) for visualizing time-aware processes or do not consider visualization issues at all. Closest to this work is the proposal made by Eder et al. regarding personal schedules [9]. In particular, a personal schedule contains all (future) activities assigned to a specific user and suggests an optimal execution order for them according some optimization strategy. Thereby, the activities of all process instances executed in the PAIS are considered, i.e., the activities of a personal schedule may originate from instances running on different process schemas.

3 Backgrounds

This section provides basic notions. Sect. 3.1 discusses fundamentals on time-aware processes, while Sect. 3.2 introduces Gantt charts.

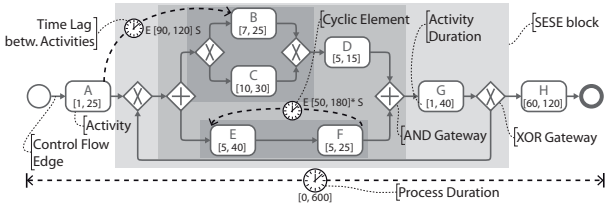

Fig. 1: Well-structured Time-aware Process Schema

3.1 Fundamentals on Time-aware Process Schema

A *process schema* is described in terms of a directed graph whose node set comprises *activities* and *gateways*. Thereby, an activity corresponds to a human task, requiring user interaction, or an automated task. In turn, gateways may be categorized as *AND/XOR*-gateways and may be used for modeling parallel/conditional branchings, and loops. (Note that the latter are expressed through XOR-gateways.) Control edges between activities and/or gateways represent precedence relations, i.e., the *control flow* of the process schema (cf. Fig. 1).

This paper, presumes that process schemas are *well-structured* [16], i.e., sequences, branchings (of different semantics), and loops are specified as blocks with well-defined start and end nodes having the same gateway type. These blocks, also known as *SESE (single-entry-single-exit)* blocks (cf. Fig. 1), may be nested, but are not allowed to overlap.

Regarding the temporal perspective, the following time patterns (TP) are considered in the following: *time lags between two activities* (TP1), *duration* (TP2) of activities and processes, and *cyclic elements* (TP9). These pattern are selected since they are the most relevant ones in practice. A full list of time patterns for PAISs as well as their formal semantics are described in [4, 5, 17].

An **activity duration** (TP2) restricts the time span allowed for executing an activity, i.e., the time span between the start and end events of the activity [5, 17]. We assume that each activity in a process schema has an assigned duration. Activity durations are expressed in terms of a time interval $[MinD, MaxD]$ with $1 \leq MinD \leq MaxD$ (cf. Fig. 1). In addition, a process schema itself may have a **process duration** (TP2) representing the allowed time span between the start and end of corresponding process instances [5, 17].

Time lags between two activities (TP1) restrict the time span between start/end events of two activities [5, 17]. Such a time lag may not only be defined between directly succeeding activities, but between arbitrary activities, presuming they may be conjointly executed in the context of the same process instance. A time lag is visualized by a dashed edge with a clock (cf. Fig. 1). The label of the edge specifies the constraint: $\langle I_S \rangle\, [MinD, MaxD]\, \langle I_T \rangle$, where $\langle I_S \rangle \in \{S, E\}$ and $\langle I_T \rangle \in \{S, E\}$ mark the events (i.e., start/end) of the source/target activity; e.g., $S[10, 20]E$ expresses that the time lag between the start of the source activity and the end of target activity shall be between 10 and 20 time units.

Cyclic elements (TP9) restrict the time span between activity instances of different iterations of a loop [5, 17]. This includes instances of the same activity

as well as different activities of a loop. Like time lags, a cyclic element is visualized by a dashed edge with a clock between the source and target activity (cf. Fig. 1). To distinguish between the two, the label of a cyclic element is annotated with a "*" next to the allowed range: $\langle I_S \rangle\, [MinD, MaxD]^*\, \langle I_T \rangle$.

3.2 Fundamentals on Gantt Charts

Gantt charts are used to visualize schedules in the context of project management [11, 10]. Generally, a Gantt chart is a bar chart that displays *activities* of a project on a time line together with their temporal relationships. Each activity has a dedicated start and end time. For example, in Fig. 2, activity A is planned to start at $t = 0$ and end at $t = 15$. The horizontal length of an activity represents its duration. Usually the average duration of activities is visualized. In particular, it is not possible to visualize the minimum and maximum duration of an activity. Directed edges between activities represent temporal relationships. To be more precise, Gantt charts support *start-start*, *end-start*, and *end-end* relationships. A *start-start* relationship expresses that both activities start at same time (e.g., B and C in Fig. 2). In turn, an *end-start* relationship indicates that an activity may start after finishing another one (e.g., A and B). Finally, activities with *end-end* relationship must be finished at same time (e.g., C and D).

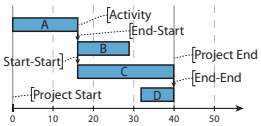

Fig. 2: Gantt Chart

4 Gantt-based Visualization of Time-aware Processes

4.1 Transforming Process Schemas to Gantt Charts

This section shows how to transform a well-structured time-aware process to a corresponding Gantt chart. Compared to traditional Gantt charts, however, time-aware process schemas are more expressive (cf. Sect. 3), e.g., considering minimum and maximum activity durations, minimum and maximum time lags between activities, exclusive choices, and loops. Consequently, Gantt charts must be extended to allow for a mapping of time-aware process schemas to them. We denote this extension as *extended Gantt* (eGantt) chart.

Activity Transformation. When transforming a time-aware process schema to an eGantt chart, each process activity A must be mapped to a corresponding eGantt activity A (cf. Fig. 3). Thereby, the length of the bar is chosen according to the minimum duration $MinD$ of A. In addition, a dashed bar is appended visualizing the maximum duration $MaxD$ of A, i.e., length of the dashed bar is $MaxD - MinD$. In case $MaxD = MinD$, the dashed bar is omitted. Overall, this allows users to easily perceive and assess temporal properties of activities.

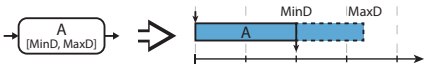

Fig. 3: Visualizing an Activity

Sequence Block Transformation. To transform a sequence of process activities to an eGantt chart, the temporal relationships between the activities must

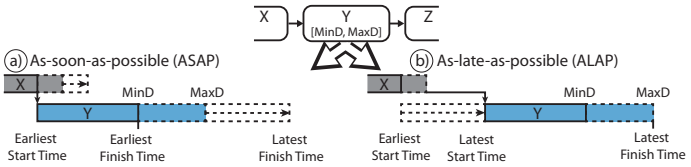

Fig. 4: Time-centric Process Appearance Alternatives

be taken into account. However, since actual activity duration is unknown at build-time no distinct position exists to place the start of succeeding activities on the time line. Even if the minimum/maximum duration of each activity is known at build-time the actual duration will be only known at run-time after completing the activity. To address this, for each process activity, four time values are calculated based on the temporal information provided by the process schema [13, 14]: *earliest start time* (EST), *earliest finish time* (EFT), *latest start time* (LST), and *latest finish time* (LFT). Note that these values are known from project planning techniques as well [18]. In particular, they provide a reference frame for the temporal properties of a process schema. Moreover, these values enable us to derive the *critical path* of a process schema which is essential for evaluating its temporal properties. Based on these values, eGantt defines two time-centric process appearances:

- *As-Soon-As-Possible* (ASAP): Each activity is assumed to be started at its earliest possible start time and to be completed after its minimum duration (i.e., at its EFT). Consequently, the succeeding eGantt activity (i.e., the respective bar) starts at its EST (cf. Fig. 4a).
- *As-Late-As-Possible* (ALAP): Each activity is assumed to be started at its latest possible start time and to be completed at its LFT. Hence, the succeeding eGantt activity starts at its LST (cf. Fig. 4b).

To be able to fully assess temporal properties of a process schema, in general, it is necessary to know both the EST and LFT of the corresponding activities. Hence, when choosing ASAP as eGantt appearance, a dashed bar with an arrow is attached to each eGantt activity visualizing its LFT (cf. Fig. 4a). In turn, when choosing ALAP as eGantt appearance, a dashed bar with arrow is placed before the respective eGantt activity visualizing its EST (cf. Fig. 4b).

eGantt activities are connected by arrows indicating their temporal relationship (i.e., precedence relations). Regarding ASAP appearance, the arrow starts at the EFT of the first and ends at EST of the second activity (cf. Fig. 4a). Concerning ALAP appearance, the arrow starts at LFT of the first and ends at LST of the second activity (cf. Fig. 4b). Due to lack of space, we focus on the ASAP appearance for the remainder of this paper.

Following this, time lags between activities need to be considered. For each time lag an arrow is added between the respective activities. Depending on the type of eGantt appearance (i.e., ASAP or ALAP), the arrow connects the EST/EFT or LST/LFT of the activities according to the kind of time lag (i.e., start-start, start-end, end-start, or end-end). Note that, time lags affecting the

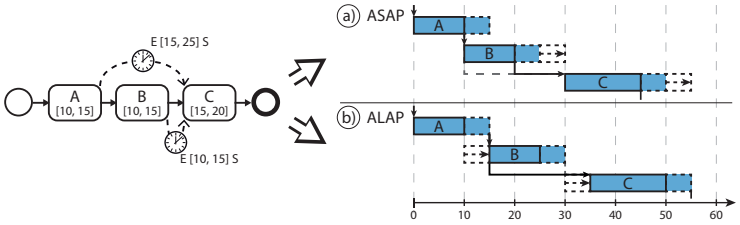

Fig. 5: Example of an Activity Sequence Visualization

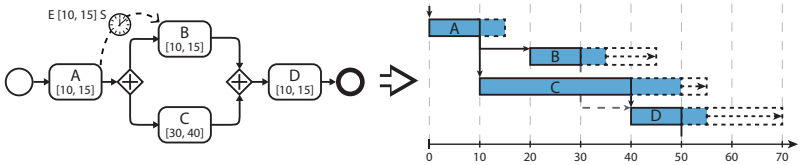

Fig. 6: Parallel Gateway Visualization (ASAP)

start or end time of an activity are taken into account through calculating respective time values (i.e., EST, LST, EFT, and LFT).

Example 1 (Activity Sequence). Consider the process schema depicted in Fig. 5. It contains three sequential activities with defined time lags between A and C as well as B and C. In particular, the time lag between B and C requires that the earliest start time of C is 10 time units after completion of B. Consequently, the EST of C must be delayed by 10 time units (cf. Figs. 5a+b). In turn, the time lag between A and C requires that C must not be started more than 25 time units after completing A. Note that this has no impact on the EST, but requires the LST of activity C to be restricted to time point 35. Regarding Fig. 5a, it is noteworthy that the time lag between A and C has no impact on the earliest start or end time of C. Particularly, the time lag is not part of any critical path.

Generally, temporal relations not part of a critical path are visualized using a dashed grey line instead of a solid one (cf. Fig. 5a). This way, the critical path, being essential for evaluating temporal properties, can be recognized by users.

Parallel Block Transformation. Regarding the transformation of parallel blocks a similar approach is taken. When using ASAP appearance, activities directly succeeding an AND-split (i.e., all branches) may be started after completing the preceding activity. However, it might become necessary to delay the start of one or more branches due to the presence of a time lag (cf. Fig. 6). Following this, the earliest end time of a parallel block is determined by the latest EFT of all branches (cf. Fig. 6). In fact, a subsequent activity must not be started before having completed all branches of the parallel block.

Conditional Block Transformation. Exclusive choices are not supported by traditional Gantt charts. As particular challenge, during run-time, only one of the branches is executed. Generally, it is therefore not possible to determine an exact EST/LST for activities succeeding a conditional block. To support exclusive choices, our eGantt charts introduce distinct *conditional containers*.

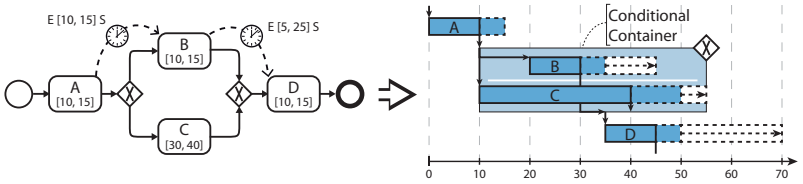

Fig. 7: Conditional Block Visualization (ASAP)

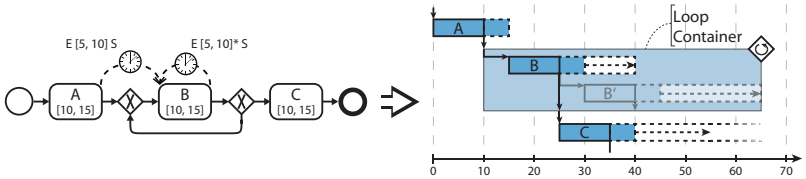

Fig. 8: Loop Block Visualization (ASAP)

These encapsulate all branches of an exclusive choice and are marked with a diamond containing an 'X'-symbol on the top right (cf. Fig. 7). Furthermore, respective branches are visually separated through a horizontal line. In the context of ASAP appearance, the activity succeeding the XOR-join is then positioned according to its earliest EST in all alternative execution paths, i.e., branches (cf. Fig. 7).

Loop Block Transformation. Like conditional blocks, loop blocks have not been considered in Gantt charts so far. Therefore, eGantt charts introduce *loop containers* for visualizing loops. In turn, these containers are marked by a diamond containing a loop symbol on the top right (cf. Fig. 7). Generally, loops may be considered as extension of an exclusive choice (see [15]): After each iteration, a decision is made whether to exit the loop or insert another copy of its loop block; eGantt charts adopt this. In particular, when visualizing a loop, a single iteration is added to the loop container in the eGantt chart. Next, a "virtual" second iteration (partially translucent) is added to indicate that the loop body may be executed again (cf. Fig. 8). The activity succeeding the loop is positioned according to the first iteration. Otherwise, it might not be possible to calculate the LST of this activity (i.e., no maximum number of iterations is available). However, to indicate the open end of this eGantt activity, the bar representing the LFT does not end, but fades out after the point of LFT calculated based on the first iteration (cf. Fig. 8). Finally, any subsequent activity is positioned according to the EST/EFT which has been used to position the eGantt activity.

4.2 Executing eGantt-based Process Appearances

When executing a time-aware process schema respective temporal constraints need to be monitored. Furthermore, users need to be informed about the progress of the process instance and its temporal properties. eGantt supports this by providing a run-time visualization (cf. Fig. 9).

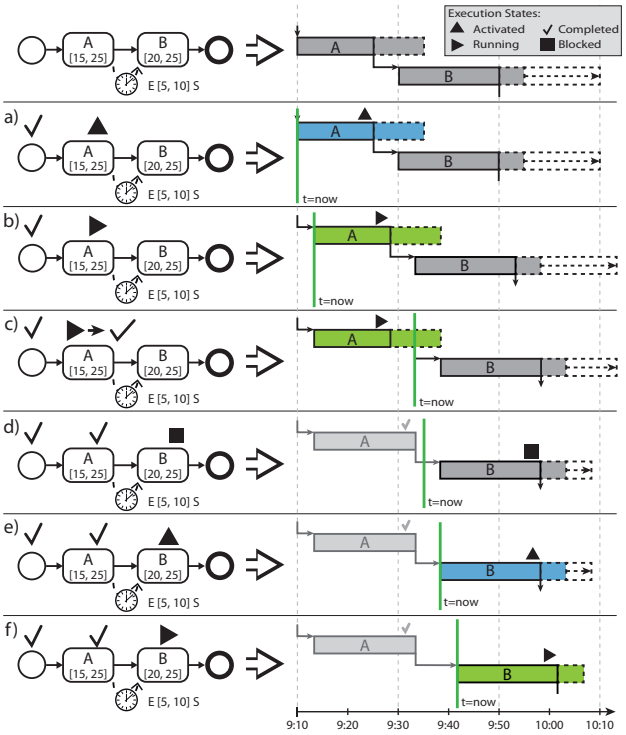

Fig. 9: Run-time Visualization of eGantt Charts

When instantiating a process schema, first of all, the corresponding eGantt chart is configured according to the current time; i.e., the time line is adapted to reflect the time the process is executed (e.g., 9:10 am in Fig. 9). Next, a *progress line* is added to represent the current point in time on the time line (cf. Fig. 9a). Finally, execution state symbols are used to visualize the state of the activities, e.g., an activity may be *activated* (but not yet *running*), *running*, *blocked* (due to a minimum time lag), or *completed*.

When executing a process instance, initially, its first activity enters execution state *activated* (cf. Fig. 9a). To reflect the current time, an activity in this state–together with the progress line–moves along the time line until it starts. Further, all other activities also move along the time line reflecting the pending start of the first activity (cf. Fig. 9b). This visualization allows users to correctly assess future execution times of subsequent activities as well as temporal properties of the process instance at any time.

As soon as the activity is started by a user, it stops moving and switches to execution state *running* (cf. Fig. 9b). At this point, the LFT is the same as the current time plus the maximum duration of the activity, i.e., the bar visualizing the LFT is removed. As soon as the activity duration exceeds the minimum duration, succeeding activities start moving again to reflect this.

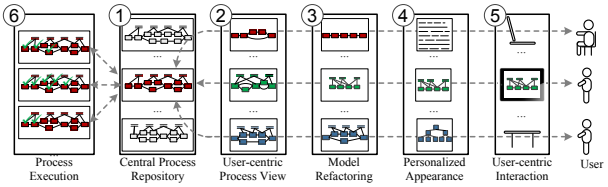

Fig. 10: proView Framework

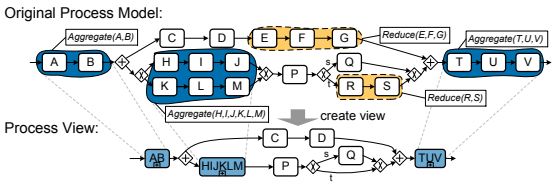

Fig. 11: Example of a Process View

When *completing* an activity, the length of its bar is adapted according to its actual execution duration. The positions of subsequent activities on the time line are the adapted according to the actual duration of the preceding activity (cf. Fig. 9c). Next, the succeeding activity is either marked as *blocked* or *activated*. An activity is blocked, if a minimum time lag has not been reached yet (cf. Fig. 9d) and the user must wait before executing it. In turn, an activity in state activated may be started by the user (cf. Fig. 9e). When reaching the end of the process instance, the eGantt chart visualizes the actual durations and execution history of the process instance.

5 Embedding eGantt charts in a Process Management Framework

We incorporated the presented eGantt charts in the *proView* framework to create personalized process schedules; *proView* aims at supporting users in interacting with large business process schemas and evolving them at a high level of abstraction [19, 20]. For this purpose, personalized and updatable process views (cf. Fig. 10, 2) are created for each user role, abstracting from large and complex process schemas stored in the central process repository (cf. Fig. 10, 1).

A process view abstracts from a large process schema by hiding non-relevant process elements (i.e., reduction operation) or by combining them to an abstracted node (i.e., aggregation operation). Fig. 11 gives an example of how such a process view is constructed in *proView* (see [21, 22] for details).

When applying view creation operations, superfluous gateways or AND branches without corresponding activities might result. Therefore, the schema of a process view is simplified through behavior-preserving refactorings (cf. Fig. 10, 3), e.g., AND-gateways of a parallel branching with only one remaining branch are removed. Furthermore, *proView* allows transforming the resulting process view

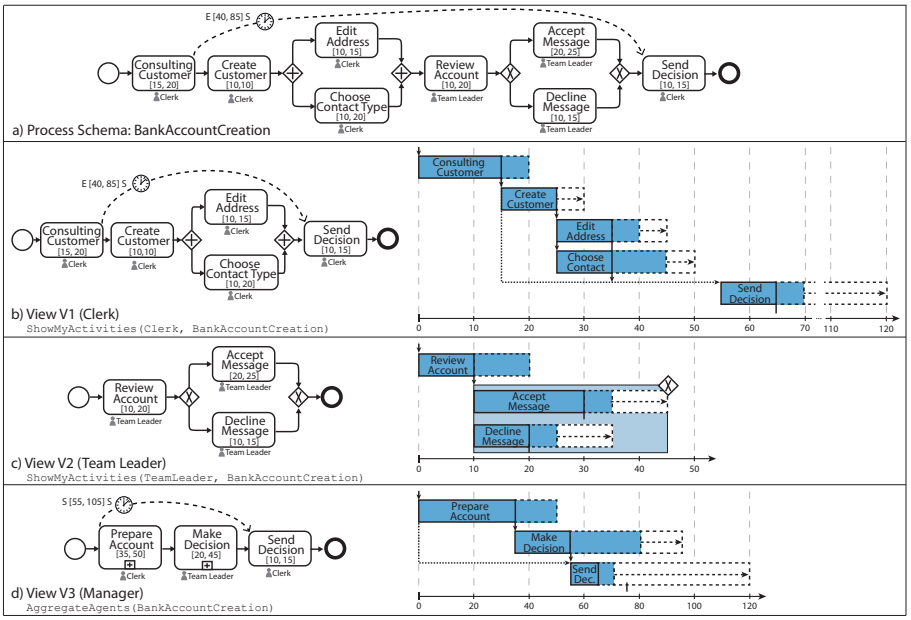

Fig. 12: Creating Personalized Process Schedules

into a personalized visual appearance (cf. Fig. 10, 4), e.g., a textual, form-based, and tree-based representation, or—as suggested in this paper—an eGantt chart. Using the resulting visual appearance (cf. Fig. 10, 5) , the user may then read or update the process schema [23, 24]. Finally, *proView* supports process execution (cf. Fig. 10, 6). In this context, run-time information is added to process views and process appearances (e.g., eGantt charts).

5.1 Creating Personalized Process Schedules

We present a two-step-method for creating a personalized process schedule visualizing a time-aware process schema for a particular user. To illustrate this method, Fig. 12 introduces a simple scenario related to a *bank account creation* process. This process involves a clerk, team leader, and manager. The clerk consults the customer and prepares the creation of the account. In turn, the team leader checks whether the account creation is accepted or declined. Finally, the clerk informs the customer about the decision. Furthermore, a manager supervises overall process execution. All activities have minimum and maximum durations indicating the time a user is expected to work on the particular activity. Further, there is a time lag expressing that *after* consultation of the customer, he gets the decision after 40 time units the earliest and 85 time units the latest. To create personalized process schedules, for each user as well as for the manager requiring an abstract overview of the process, the following two steps are performed:

Step 1. First, *role-specific process views* are created for the given process schema to abstract from particular process aspects. In our context, three process

views are required. High-level view creation operation `ShowMyActivities` creates a process view for roles *clerk* (cf. Fig. 12b) and *team leader* (cf. Fig. 12c) eliminating activities not performed by them. Furthermore, operation `AggregateAgents` combines SESE blocks performed by the same user to one abstracted activity (cf. Fig. 12d). Note that the time lag between activities *Consulting Customer* and *Send Decision* is maintained for process views *V1* and *V3*. In particular, view operation `AggregateAgents` needs to adjust the time lag since the original source activity is aggregated. For this, the source of the time lag is adapted to the start of the aggregated activity *Prepare Account*. View V2 does not include the time lag anymore since it is not relevant for the respective user.

Step 2. The *eGantt chart transformation* is performed by applying the transformations described in Sect. 4 to the created view schemas. Note that the gap before activity *Send Decision* in view V1 is due to combination of the time lag and the elimination of the team leader's activities; i.e., before executing this activity, the clerk must wait until preceding activities of the team leader are completed and the minimum time lag is reached. Since view V2 has no such temporal relationships, no gap is visualized in the eGantt chart. Finally, the eGantt chart of V3 shows the aggregated SESE blocks of the users involved. V3 and its eGantt chart appearance give a comprehensive and abstract overview of the temporal information captured in the time-aware process schema *Bank Account Creation*.

6 Summary and Outlook

We introduced personalized process schedules based on time-aware process schemas. For this, time-aware process schemas are abstracted using high-level view-creation operations. Based on personalized process views, the transformation to extended Gantt charts is described. In particular, our process visualization approach enables users to easily understand the temporal perspective of process schemas. In future work, we evaluate proposed eGantt charts and fully implement them in the *proView* framework including its execution environment. Further, we will develop time-centric view operations as well (e.g., to only show activities executed within a certain time frame). Finally, temporal information shall be extracted from process logs using process mining techniques. In turn, this information will be used to refine respective eGantt charts.

Acknowledgement. The authors would like to acknowledge and thank Martin Sommer for providing first results on this topic.

References

1. Reichert, M., Weber, B.: Enabling Flexibility in Process-aware Information Systems - Challenges, Methods, Technologies. Springer (2012)

2. Lenz, R., Reichert, M.: IT support for healthcare processes - premises, challenges, perspectives. Data & Knowledge Engineering **61**(1) (April 2007) 39–58
3. Müller, D., Herbst, J., Hammori, M., Reichert, M.: IT support for release management processes in the automotive industry. In: Proc. BPM'06. (2006) 368–377
4. Lanz, A., Weber, B., Reichert, M.: Workflow time patterns for process-aware information systems. In: Proc. BPMDS'10 Workshop. (2010) 94–107
5. Lanz, A., Weber, B., Reichert, M.: Time patterns for process-aware information systems. Requirements Engineering (2013)
6. Combi, C., Gozzi, M., Juarez, J.M., Oliboni, B., Pozzi, G.: Conceptual modeling of temporal clinical workflows. In: Proc. Int'l Symp. TIME'07, IEEE (2007) 70–81
7. Eder, J., Panagos, E., Rabinovich, M.: Time constraints in workflow systems. In: Proc. Int'l Conf. CAiSE'99. Volume 1626 of LNCS., Springer (1999) 286–300
8. Dadam, P., Reichert, M., Kuhn, K.: Clinical workflows - the killer application for process-oriented information systems. In: Proc. Int'l Conf. BIS'00. (2000) 36–59
9. Eder, J., Pichler, H., Gruber, W., Ninaus, M.: Personal schedules for workflow systems. In: Proc. BPM'03. Volume 2678 of LNCS., Springer (2003) 216–231
10. Maylor, H.: Beyond the gantt chart: Project management moving on. European Management Journal **19**(1) (2001) 92–100
11. Clark, W.: The Gantt Chart. The Ronald Press Co. (1922)
12. Kosara, R., Miksch, S.: Metaphors of movement: a visualization and user interface for time-oriented, skeletal plans. Artif Intell Med **22**(2) (2001) 111 – 131
13. Bettini, C., Wang, X.S., Jajodia, S.: Temporal reasoning in workflow systems. Distributed and Parallel Databases **11**(3) (2002) 269–306
14. Eder, J., Panagos, E.: Managing time in workflow systems. In Fischer, L., ed.: Workflow Handbook 2001. Future Strategies Inc. (2000) 109–132
15. Combi, C., Gozzi, M., Posenato, R., Pozzi, G.: Conceptual modeling of flexible temporal workflows. ACM Trans. Auto. & Adapt. Syst. **7**(2) (2012) 19:1–19:29
16. Kiepuszewski, B., ter Hofstede, A., Bussler, C.: On structured workflow modelling. In: Proc. CAiSE'00. Volume 1789 of LNCS., Springer (2000) 431–445
17. Lanz, A., Reichert, M., Weber, B.: A formal semantics of time patterns for process-aware information systems. TR UIB-2013-02, University of Ulm (2013)
18. Lockyer, K.G., Gordon, J.: Project Management and Project Network Techniques. Prentice Hall (2005)
19. Kolb, J., Kammerer, K., Reichert, M.: Updatable process views for user-centered adaption of large process models. In: Proc. 10th Intl. Conf. on Service Oriented Computing (ICSOC'12). (2012)
20. Kolb, J., Kammerer, K., Reichert, M.: Updatable process views for adapting large process models: The proview demonstrator. In: Proc. BPM'12 Demo Track. (2012)
21. Reichert, M., Kolb, J., Bobrik, R., Bauer, T.: Enabling personalized visualization of large business processes through parameterizable views. In: Proc. Symp. Appl. Comp. (SAC'12). (2012)
22. Kolb, J., Reichert, M.: Data flow abstractions and adaptations through updatable process views. In: Proc. Symp. Appl. Comp. (SAC'13). (2013) (to appear)
23. Kolb, J., Rudner, B., Reichert, M.: Towards gesture-based process modeling on multi-touch devices. In: Proc. 1st Int'l Workshop on Human-Centric Process-Aware Information Systems (HC-PAIS'12). (2012) 280–293
24. Kabicher-Fuchs, S., Rinderle-Ma, S., Recker, J., Indulska, M., Charoy, F., Christiaanse, R., Dunkl, R., Grambow, G., Kolb, J., Leopold, H., Mendling, J.: Human-centric process-aware information systems. CoRR **abs/1211.4** (2012)