Evaluating Compliance State Visualizations for Multiple Process Models and Instances

Manuel Gall[1,2] and Stefanie Rinderle-Ma[3]

[1] Austrian Center of Digital Production, Vienna, Austria
[2] University of Vienna, Faculty of Computer Science, Vienna, Austria
manuel.gall@univie.ac.at
[3] Technical University of Munich, Department of Informatics, Garching, Germany
stefanie.rinderle-ma@tum.de

Abstract. Business process compliance refers to the formalization, enactment, verification, and monitoring of constraints for one or multiple process models and one or multiple process instances. Such complex compliance scenarios crave for visualization support that fosters traceability and understandability during design and runtime. It must be clear, for example, which processes and process instances are subject to which compliance constraint and, especially during runtime, which compliance state (e.g., satisfied or violated) is active. This paper analyzes existing visualization approaches for compliance-related information and demonstrates their usability and feasibility through a prototypical implementation and the application to a logistics scenario. The focus is on constraints that span across multiple processes and process instances. The preferred visualizations are then implemented in a real-world process scenario from the manufacturing domain and evaluated through in-depth interviews with three stakeholders. The interviews narrow down the results of the technical evaluation, indicating that *Color* is best suited for obtaining a quick overview and *Text* for in-detail analysis of compliance states.

Keywords: Business Process Compliance, Compliance Visualization, Compliance Traceability, In-Depth Interviews

1 Introduction

Business process compliance is expensive for companies, but the costs for non-compliance can be far higher [2]. Due to the COVID pandemic and digitalization needs companies gear up on compliance spendings, i.e., *"legal technology budgets will increase threefold by 2025"* [24]. Business process compliance means to formalize, enact, verify, and monitor compliance constraints stemming from, e.g., regulatory documents, in connection with process models and process instances [15]. Compliance scenarios can become complex due to the following reasons:

1. Compliance scenarios may comprise a multitude of process models, process instances, and diverse constraints. [22], for example, describes a scenario for one organization in the higher education domain with 108 process models,

5831 activities, and 375 constraints. Compliance constraints can refer to none, a subset, or all process models and instances (we call a constraint *active* on a model or instance if it refers to it). Therefore *compliance traceability* is a desired goal, i.e., it has to be clear what belongs together.
2. Constraints may span multiple process models and instances, so called instance-spanning constraints (ISC), e.g., for bundling/unbundling of cargo [7]. Multiple ISC can be active on the same process instance. This might result in conflicting visualizations on a single activity.
3. Compliance states may have to be checked during design time and runtime [15]. Runtime checks include the distinction of life cycle states for process activities such as active or complete [18] and compliance constraints, i.e., pending, satisfied, and violated [18]. These states have to be visualized in a way that the states can be distinguished and thus support the understandability of the whole visualization.

Motivation (i) – (iii) shows overseeing and assessing the compliance states of all process models and instances for all constraints can become a cumbersome and arduous task for process analysts and compliance officers. Hence, with a focus on ISC (ii), this work tackles the following research questions:

RQ1 Which requirements need to be satisfied by an ISC visualization?
RQ2 How can ISC be visualized on running process instances?
RQ3 Which ISC visualization is best suited for assessing compliance states?

We will follow the design science research methodology [26] as follows: Requirements to be met by an ISC visualization are harvested from constraint management literature (cf. Sect. 2). Artifacts for ISC visualization are created based on literature from constraint visualization, information visualization, and graph visualization (cf. Sect. 3). Feasibility and coverage of all ISC visualizations are evaluated against the requirements based on a prototypical implementation (cf. Sect. 4). The findings are then further evaluated based on in-depth interviews with stakeholders in the context of a real-world scenario from manufacturing (cf. Section 5). Section 6 provides a conclusion.

2 Visualization Requirements

This section collects and groups visualization requirements for business process compliance. Grouping the requirements facilitates the comparison of existing approaches in this area. The grouping strategy is developed based on literature [8,9,20] and consists of the following four perspectives: process models, process instances, ISC, and ISC instances. The visualization requirements are collected from a selection of constraint management literature [12,13,16,18] and consider the constraint lifecycle states *pending*, *satisfied*, and *violated* [18] . The goal of collecting and grouping the visualization requirements is to identify how many of these requirements are satisfied by an existing visualization approach. Based on this assessment, recommendations for visualizing business process compliance scenarios can be derived. Visualization requirements are ($\mapsto$ **RQ1**):

– *Process Perspective*
  1. It should be possible to identify multiple active ISC in any state on one or multiple process models. [16,18]
– *Process Instance Perspective*
  2. For each activity it should be possible to identify the currently active constraint states. [13,16]
  3. For each activity it should be possible to identify the currently active ISC. [13,16]
  4. For each activity it should be possible to identify the active ISC and their constraint states. [13,16]
– *ISC Perspective*
  5. It should be possible to identify all ISC instance states based on the ISC visualization. [16]
  6. For each ISC instance it should be possible to identify on which activity they are currently active.[16]
– *ISC Instance Perspective*
  7. It should be possible to identify the process on which an ISC instance is active on. [16]
  8. It should be possible to identify the process instances on which an ISC instance is active on. [16]
  9. It should be possible to identify the activities on which an ISC instance is active on. [12,16]

Figure 1 (top) depicts a real-world manufacturing process model from EVVA Sicherheitstechnologie GmbH[4]. A pallet transports parts to the station, where an employee scans the product code. If the scan is successful, the product data is loaded. After loading, the employee is shown a step-by-step instruction on how to assemble the product. At the bottom, left of Fig. 1 two assembly lines are depicted with associated process instances. Assume an ISC requires that currently more than 2 instances are waiting for a pallet (depicted as ISC at bottom, middle). One ISC instance is created and active per assembly line (bottom, right). The colors of the ISC instances, i.e., yellow and red, reflect their compliance states, i.e., *pending* and *violated*. The corresponding ISC state is consequently visualized using both colors. For this example, we can say that *visualization requirement 5* is fulfilled.

3 Visualization Approaches

The goal for compliance visualization is to inform the user about the current state of each ISC on the process model and process instances of interest [18]. We define **ISC traceability** as the user's ability to identify an ISC in a specific state on multiple instances from multiple processes, i.e., the visualization can cover states of a single and multiple ISC at a time.

[4] `www.evva.com`

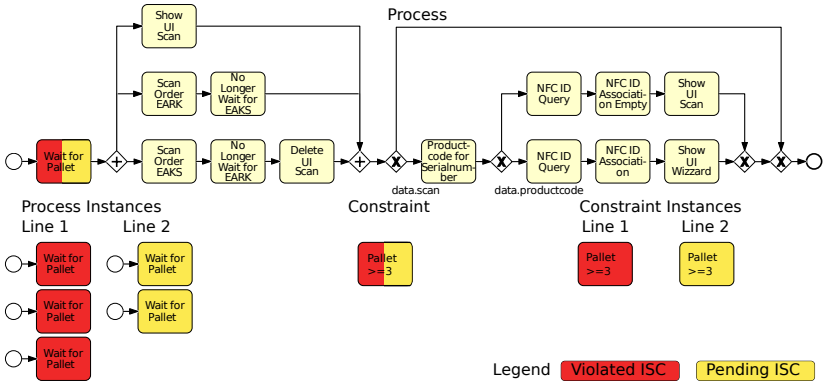

Fig. 1: Real-world manufacturing process: using color for ISC visualization.

In order to answer **RQ2**, we collected approaches from constraint visualization [1,4,16,17,18,25], information visualization [3,19], and graph visualization [23]. The collected approaches focus on intra-instance constraint visualization and will be transferred to an ISC context for this work. Hence, we are confident that the eventually selected approach will be suitable for visualizing ISC as well as intra-instance constraints.

Figure 2 depicts the considered visualization approaches along three visualization tasks, i.e., visualizing *one constraint state*, *three constraint states*, and *three constraint states multiple ISC*. In detail, the first column shows one active constraint state, the second one –from left to right– constraint states *satisfied*, *pending*, and *violated*, and the third column multiple active ISC in all three constraint states.

In order to visualize multiple instances from multiple processes we use the 3D framework presented in [10]. Using a 3D setup allows us to use augmented reality (AR) and virtual reality (VR) capabilities in future development. Most of the approaches could be used in the same way in a 2D scenario. An exception are visualizations utilizing the additional axis, e.g., the *orientation* approach covers all three constraint states as rotating along the X, Y, and Z axis is possible. Using a 2D setting would not allow for three states to be covered by *orientation*. Assuming a 3D visualization, each of the cubes in the cells of Fig. 3 represents a process activity.

Some of the visualization approaches such as *size* do not allow to depict three constraint states on a single activity. Others such as *orientation* do not allow to depict multiple states of the same type. In such cases the associated cell is left empty. In the following, we discuss the different visualization approaches.

Symbols are represented near activities and ISC. By using the same symbol multiple times connections between perspectives, e.g., process instance and ISC instance can be drawn [16]. By utilizing a symbol's visual attributes, i.e., shape and color additional information such as the constraint state can be encoded [25]. Multiple symbols close to an activity represent different ISC.

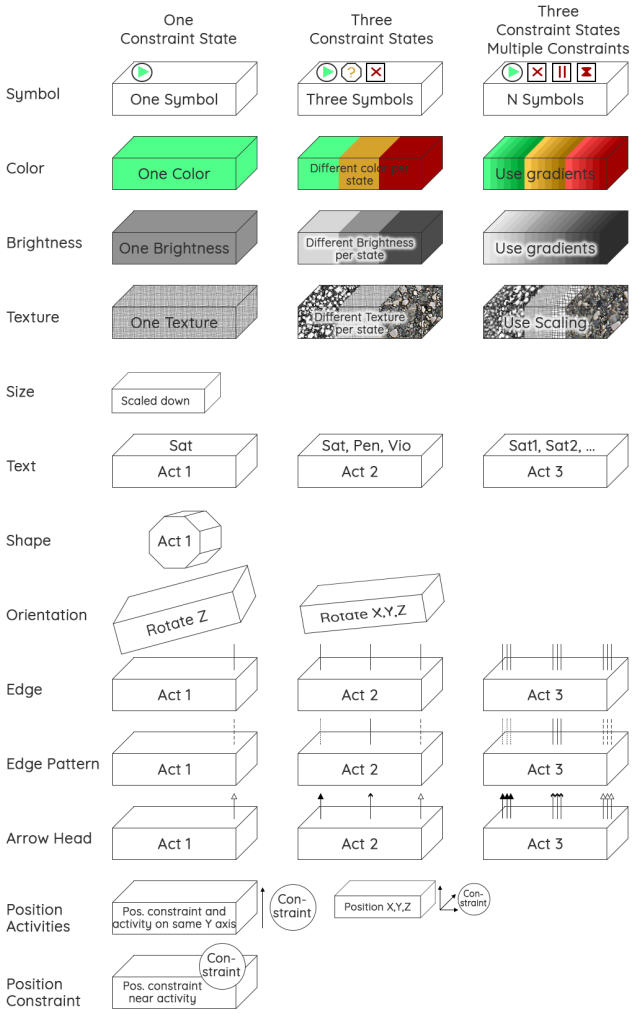

Fig. 2: Visualization approaches for constraint states on process activities.

Color enables a range of applications. [1,4,16] use colors to express one of the constraint states for an activity. Colors can be used to further specify which ISC is currently active on a certain activity by using the same color [6,17]. To visualize multiple active ISC for a single activity color ranges [23] can be used. Using color ranges from, e.g., light green to dark green, allows to differentiate between multiple states.

Brightness [3] can be used in a similar way as *Color*. Multiple constraint states can be expressed by different brightness ranges.

Texture [3] represents the perceived surface of an object. Textures range from line drawings to colored images. To represent constraint states of an activity

different textures can be used. Texture can encode information [11] by using color channel, tilling, smoothness, and many other attributes. Due to these different encodings a texture is suitable to represent different constraint states with ease.
Size: An activity and its associated ISC can be displayed using the same size. For discrimination to other activities, the size can be varied [19]. Specifically for visualizing constraint states, three groups can be defined, i.e., small for *pending*, medium for *verified*, and large for *violated*. Similar to the *Color Range*, each group can use a range for scaling, e.g., small uses a scaling of 30%-70%, medium 80%-120%, and large 130%- 170%.
Text is mostly used to complement other visual styles such as *Colors* [18]. *Text* can be used without other visualizations [12,19] for constraint state representation. *Text* can be used to indicate if an ISC is currently active on an activity by writing the constraint state close to the activity or writing the name of the active ISC near the activity.
Shape: Activities [3] and the associated ISC can be set to the same unique shape to represent their association. Specific shapes such as triangular for state pending, can be used to express constraint states.
Orientation: By setting the same orientation [3] of activities and ISC, their association can be displayed. Different axis can be used to depict certain constraint states. For example, rotations on the X-Axis might equal to constraint state *satisfied*.
Edge: For each ISC, a directed edge [19] is created towards the activity. By placing the edge on fixed positions different constraint states can be encoded, e.g., front means *satisfied*, middle means *pending*, and back means *violated*.
Edge Pattern: For each edge it is possible to change the pattern [19] and to integrate information into the pattern. These patterns can be used for constraint state representation.
Arrow Head: For each edge it is possible to change the arrow head [19] and to integrate information into it. This works basically the same way as *Edge Pattern*.
Position [3] can be applied in two ways, i.e., *Position the Activities* or *Position the Constraint* [4,12,25]. *Positioning the Activities* moves the activity and ISC on the same axis position, for example, the same unique Y-Axis position. *Positioning the Constraint* positions the ISC instance near the activity bound by the ISC instance.

4 Implementation and Feasibility

The goal of the technical evaluation is to assess which visualization requirements from Section 2 are met by which visualization approaches (**RQ3**). For this, the visualization approaches –covered by a prototypical implementation– are applied to a set of four real-world ISC [21] in five logistics scenarios.

4.1 Evaluation Setup

According to [7], ISC can be classified along two properties, i.e., *context* and *modeling*. Context signifies if an ISC spans multiple processes or instances. Mod-

eling refers to which process attributes such as time, data, and resource an ISC refers to. The possible combinations of context and modeling are reflected by the following four real-world ISC1 – ISC4 from the logistics domain (from [21]):
• Single context (one process, multiple instances), single modeling requirement (destination): *"For cargo distributed over several trucks, all cargo must arrive in the same destination."* (↦ **ISC1**)
• Single context (one process, multiple instances), multiple modeling requirements (date, customer): *'There should not exist more than 3 instances of post office delivery such that a specific input parameter (say date) is the same and the post office is also the same."'* (↦ **ISC2**)
• Multiple context (multiple processes, multiple instances), single modeling requirement (time): *"The optimal case is all deliveries are on-time or 100% of on-time delivery. If the percentage of on-time delivery drops to 80%, it is considered as critical."* (↦ **ISC3**)
• Multiple context (multiple processes, multiple instances), multiple modeling requirements (priority, time): *"Prioritization and dynamic handling of cargo by cargo-vehicle interaction to ensure high priority cargo item precedence over low priority items."* (↦ **ISC4**)

Figure 3 depicts two artificial logistics process models *partner ordering* and *post office delivery*. Assume that the ISC1 – ISC4 are imposed on them and/or the process instances created based on the models.

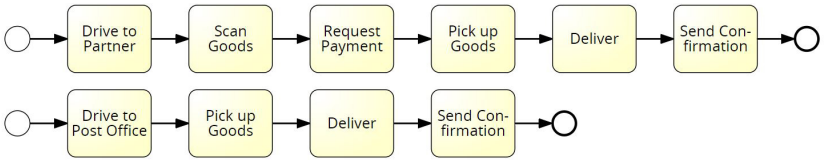

Fig. 3: Logistics process models: partner ordering and post office delivery.

Figure 4 gives an overview of the process models and instances (cf. Fig. 3) after completing the following scenarios. For constraint syntax we refer to [9].
• **Scenario1:** Partner(A) ships goods requiring two trucks (2 instances), both arriving at the same destination.
• **Scenario2:** Partner(B) ships goods requiring three trucks (3 instances). One of the trucks does not arrive at the desired location.
• **Scenario3:** A post office requires four deliveries (4 instances) within one day.
• **Scenario4:** Due to a massive traffic jam some deliveries (partner(A) instances 1 and 2, post office instances 2 and 3) are not on time.
• **Scenario5:** Partner A and Post Office delivery share the same destination. Partner A's delivery has higher priority and therefore should arrive first.

ISC1 – **ISC4** and **Scenario1** – **Scenario5** are evaluated with the following constraint states. **ISC1** is executed two times, at first, for **Scenario 1** where the ISC state evaluates to satisfied. Secondly, for **Scenario2** where the ISC state evaluates to violated. **ISC2** is executed once and violated by **Scenario3**. **ISC3**

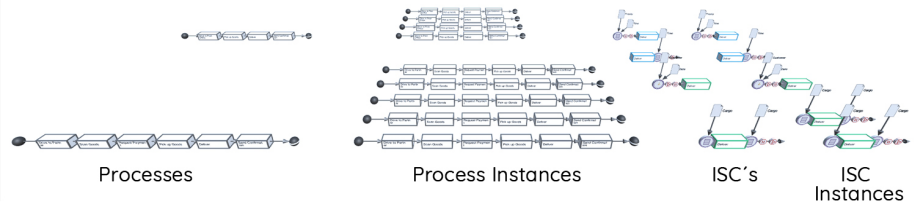

Fig. 4: Implementation overview from left to right: process models, process instances, ISC models, ISC instances

is executed once and violated due to **Scenario4**. **ISC4** is executed once and violated by **Scenario5**.

4.2 Implementation

The visualization approaches collected in Section 3 are implemented in a prototype[5]. All scenarios are evaluated with all ISC active at the same time for two reasons. (a) In a real-world setting typically multiple processes and ISC are active. (b) We aim at a better understanding on how the visualization approaches handle multiple active ISC visually interfering with each other. We used the 3D process model visualization approach [10] created in Unity3D as foundation. For using the prototype, a tutorial was created. In this tutorial, one scenario (*Partner(B) ships goods*) is executed and an ISC is highlighted to gain insight on how the process execution is done. The tutorial allows to understand an ISC visualization approach based on a simple example corresponding to the previously introduced scenarios and ISC.

For a better comparison of the visualization approaches we chose to visualize them after all scenarios are completed. By using the next button it is possible to change between the visualization approaches.

Figure 5 depicts the *Color* visualization for **Scenario2** and **ISC1**. It illustrates an observation made during prototype development. All steps except for the first one depict two different approaches for process instance visualization. On the upper half of each step the process instances depict all process activities. On the bottom half of the steps the process instances only depict complete and running activities. For the sake of an easier understanding and to show that even such little visual decision can have a big impact on the understanding of an ISC visualization approach we opted to present both process instance visualizations.

As the activity labels are barely readable, the focus is put on the visual representation of the *Color* approach and how the colors are represented in all four perspectives. Figure 5 shows the following steps.

- *Design phase:* Visualization of process model and ISC.
- *Runtime process instance creation:* 3 instances spawned next to the process.

[5] `https://cviz.crowndefense.at/`

• *Runtime constraint instance creation:* When the first process instance arrives at activity *Deliver*, a new ISC instance with state pending (yellow) is spawned.
• *Runtime constraint instance evaluation:* Little dots above the process instance visualization depict that two instances moved to the next activity. The state still remains in status pending as these instances arrived at the same destination.
• *Runtime constraint instance evaluation:* The goods for the remaining instance were delivered. These goods were delivered to a different destination and therefore violating the ISC and resulting in a change of color to red.

This example depicts a single scenario with one ISC. The prototype features multiple scenarios with multiple ISC.

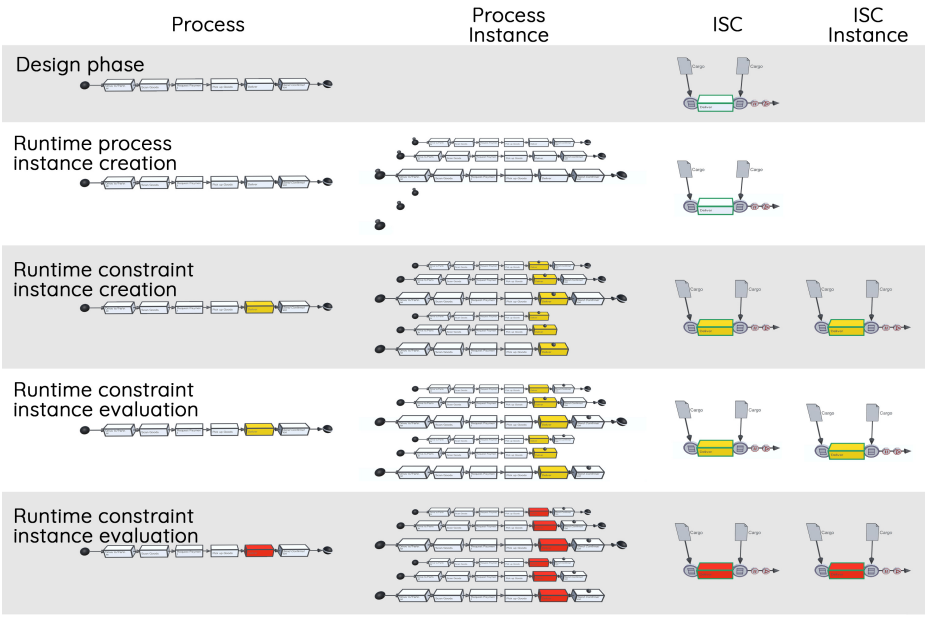

Fig. 5: Step by step example for ISC visualization with colors. Process instance activities are created during runtime. For labels see Fig. 3.

4.3 Preselection

Table 1 gives an overview on the support for visualization requirements (columns 1-9) (cf. Section 2) per visualization approach (rows) (cf. Section 3). *Position* visualization approach yields the most surprising result for both, *Position Activity* and *Position Constraint*. Before the evaluation, we thought *Position* will be one of the top choices as it is used for intra-instance constraint visualization [12,25]. However, the biggest downside of *Position Constraint* is that it cannot reflect that ISC span multiple process instances without any modifications such as duplicating the ISC. The *Position Activity* approach yields better results compared to *Position Constraint*. The approach supports up to three ISC, if the ISC

inherit different constraint states. We can position activities on the X,Y, and Z-Axis and use the same axis for the constraint states, e.g., the Y-Axis depicts a violated ISC. We would like to state that we are limited to 3 axes. Thus, if the constraint lifecycle would be extended, this approach cannot cope with the extensions. This limitation is the same for the *Orientation* approach. Currently, the *Orientation* approach rotates around all three axes. In case of a constraint lifecycle extension, some states could not be supported.

Visualization	1	2	3	4	5	6	7	8	9
Symbol	✓	✓	✓	✓	✓	✓	✓	✓	✓
Color	✓	✓	✓	✓	✓	✓	✓	✓	✓
Brightness	✓	✓	✓	✓	✓	✓	✓	✓	✓
Texture	✓	✓	✓	✓	✓	✓	✓	✓	✓
Size	□	□	□	□	□	□	✓	✓	✓
Text	✓	✓	✓	✓	✓	✓	✓	✓	✓
Shape	□	□	□	□	□	□	✓	✓	✓
Orientation	■	■	■	■	■	■	✓	✓	✓
Edge	✓	✓	✓	✓	✓	✓	✓	✓	✓
Edge Pattern	✓	✓	✓	✓	✓	✓	✓	✓	✓
Edge Head	✓	✓	✓	✓	✓	✓	✓	✓	✓
Position Activity	■	■	■	■	■	■	✓	✓	✓
Position Constraint	-	-	✓	-	-	-	-	□	-

Table 1: ✓full support, ■ support for 3 different constraint states, □ support for 1 constraint state, - not supported.

Size and *Shape* only support one constraint state as using multiple sizes and shapes at the same time would change the semantics. For example, an activity is violated and satisfied at the same time. Then one state is represented by a large cube and the other by a small cube. If the mean is used the activity could be displayed with a medium sized cube. However, a medium sized cube could have different semantics. The same problem occurs for *Shape*. Different shapes could be merged, but the merging blurs the semantics.

4.4 Discussion of Selected Approaches

The evaluation has focused on a technical and quantitative point of view, so far, and has been used to narrow down the amount of visualization approaches for constraint visualization and ISC traceability.

Based on the results from Table 1, we keep the following approaches for discussion: *Symbol*, *Color*, *Brightness*, *Texture*, *Text*, *Edge*, *Edge Pattern*, and *Edge Head*. We narrow down this list by removing *Brightness* as *Brightness* can be expressed 1:1 by *Color*. From the three different edge approaches, we will keep the *Edge Pattern* approach as it subsumes *Edge*. We will also remove *Edge Head* as the visualization only shows the constraint state near the activity. *Edge Pattern* by contrast gives a more general overview while still being able to show

the states near the activity. The remaining visualizations are *Symbol*, *Color*, *Texture*, *Text*, and *Edge Pattern*. *Color* and *Texture* are fairly similar. However, we will keep both for the discussion as one does not subsume the other.

We identified two usage scenarios: (i) getting a quick overview of all ISC (Requirements 5-9); (ii) looking into specific activities (Requirements 1-4). Based on our prototype we will give recommendations which approach to use for which scenario in the sequel.

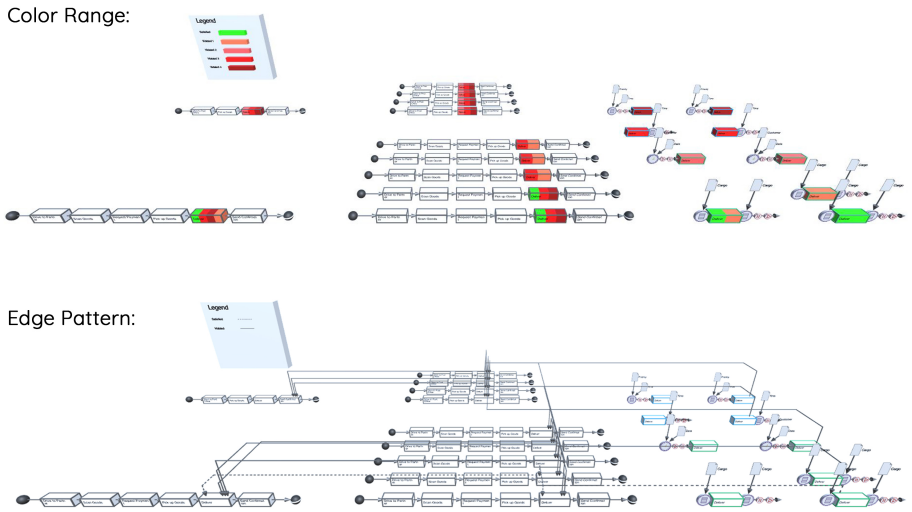

Fig. 6: Screenshot of prototype depicting all scenarios and ISC using *Color* on top and *Edge Pattern* on bottom.

Quick Overview: Our prototype shows for getting a quick overview *Symbol* and *Text* are not useful as they become too tiny to be identified on larger zoomed out graphs. *Edge Pattern* could provide potential insight into large graphs. However, ISC traceability is no longer supported as the visualization becomes cluttered with edges. Therefore, for getting a quick overview, we recommend *Color* and *Texture*. Figure 6 depicts the *Color* approach from the prototype. Even when zoomed out it is clear where the violations are located. ISC tracability is given as it is possible to see which ISC instance is connected to which activity. Figure 6 depicts the *Edge Pattern* approach on the bottom where an overview can be gained, e.g., how many activities are violated, but ISC traceability is nearly impossible.

In-Detail Analysis: *Edge Pattern* cannot be recommended for in-detail analysis as ISC traceability is not given. With a desktop environment it seems hard to follow a specific edge, especially when there are many edges on the screen. This could be different in AR/VR environments as the movement is more natural with head mounted displays. *Color* and *Texture* are well suited for getting an

overview. However, for an in-detail analysis they perform in a mediocre way. For both approaches it is hard for the user to mentally align an ISC instance to a specific color or texture. On the left side of Fig. 7, for example, activities are shown, on the right side, all ISC instances.

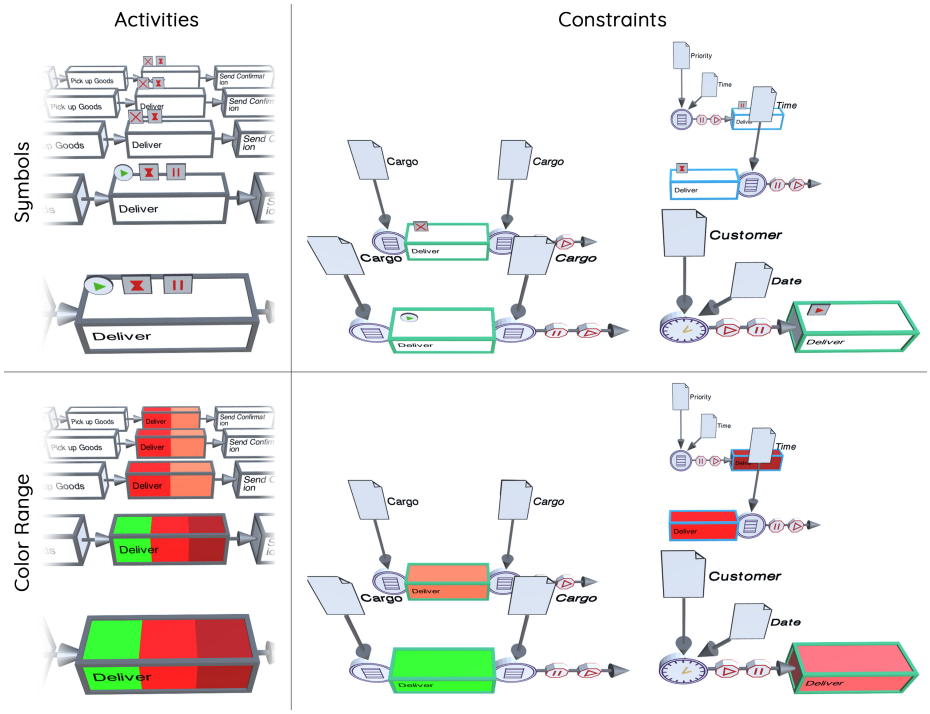

Fig. 7: Screenshot of prototype demonstrating in-detail analysis by depicting activities and ISC using *Symbol* and *Color* visualizations.

Based on the implementation, we can recommend *Text* and *Symbol* as both allow easier mental association of ISC and activity. We want to extend a bit more on these two visualization approaches as they have one critical advantage for in-detail analysis compared to the other approaches. The user can change the *Symbol* or *Text* for a more fitting representation utilizing better understanding and recognition of the ISC. All other approaches such as *Color* or *Edge Pattern* allow for similar changes, but their recognition would not be as good as *Symbol* or *Text*. For example a timed ISC (i.e., a ISC that refers to some temporal information of the process models and/or process instances) could use an hourglass symbol or be labeled with the text "timed". But what color or edge pattern could be chosen to represent a timed ISC?

As final recommendation we suggest the use of *Color* and *Texture* for getting an overview of constraint states and *Text* and *Symbol* for in-detail analysis of ISC.

5 In-Depth Interviews

The goal of the in-depth interviews is to identify new insights on ISC visualization and to compare the results with the technical evaluation provided in Section 4. For conducting the semi-structured, open-ended, in-depth interviews we followed the guidelines of Boyce and Neale [5].

Design and Methodology: The interviews were conducted with three stakeholders from the electronic montage unit at EVVA and CDP[6]. They shared the manufacturing process currently running in their production facility (cf. Fig. 1). During visits to their production facility and online meetings the process and potential risks were discussed. This helped to reduce language barriers and get to know each other such that both parties feel comfortable during the interviews [14]. We agreed to not visualize the actually running processes as they could take different paths every time and the interviews could be biased in a certain way. Instead based on the process model and annotated data process instances were simulated using CPEE(`cpee.org`) such that the process always uses the same path. We used ISC that have already occurred during process execution before such as failure of software affecting all processes and instances, failure of hardware affecting a specific station, and possible detection of hardware failure due to instance spanning data. So far these ISC have been detected by employees during production phase or testing phase.

Interviews were held online from 02.03.2021 to 04.03.2021. Besides recording the interviews, notes were taken during the interviews to allow for summarization and probing. For each interview we instantiated four instances representing four stations within the production facility. After instantiation the used engine transmits the instance information live to our visualization approach. We visualized the process, instances, ISC, and ISC instances in the same way as shown in Fig. 4. Before the interviews started an overview of all representations was provided to the stakeholders. During the interviews, we followed the prepared questions outlined at `https://bit.ly/3t7qxgE`. However, questions could be shifted or omitted depending on the respondent's answer to previous questions in order to, e.g., gain more details.

The recordings were transcribed and irrelevant phrases eliminated. The interviews were translated from German to English. Afterwards the transcripts were sent to each of the stakeholders for confirmation. They all replied that the transcript is valid and represents the interview. In preparation for the discussion we identified key topics, e.g., *Overall Best*, *Combination*, *Constraint States*, *Critical*, *Worst Approach*, *Overview*, *In-Detail Analysis*, *Presentation*, and *AR/VR* and applied color coding to verify that all information was captured[7].

Results and Discussion: Color is regarded as the *Overall Best* approach for ISC visualization. *Color* by itself is suitable for visualizing *Constraint States* and for visualizing traceability between processes/instances and ISC/ISC instances. For all stakeholders it is important to find violated ISC rather quickly. They

[6] `https://acdp.at/`

[7] Color coding available here `https://bit.ly/3etUPX2`

suggest to use a *Combination* of different visualizations to allow for quick violation finding and traceability. The stakeholders were divided between different approaches, e.g., *Shape*, *Rotation*, and *Scale* for visualizing that a violation happened. To add traceability they suggest the usage of either *Color* or *Symbol*. This result is especially interesting as it differs from our technical evaluation. Due to their limited expression we did not consider *Shape*, *Rotation*, and *Scale* any further. To use those visualization approaches as indicator that something happened is an interesting observation.

Textures are viewed *Critically* by the stakeholders. On the one side they emphasize that *Textures* could convey more information compared to color. On the other side they see problems with readability of text, the need for a legend and a longer training period. The stakeholders rarely mentioned the approaches *Position Activities*, *Position Constraint*, and *Brightness*.

Every stakeholder mentioned *Edge* as the *Worst Approach* as it clutters the visualization, particularly for complex processes and instances. Traceability is not given as in the following response: *"I went with the finger over the screen and got nowhere."* This result is inline with the technical evaluation and we will not consider edges for ISC representation.

For getting an *Overview* the stakeholders favor *Color*. For conducting an *In-Detail Analysis* they are divided between *Symbols* and *Text*. *Text* was favored because of the ability to express complicated information in a compact form: *"You can think up any text you want and display it without circumstances."* For *In-Detail Analysis*, *Text* is suitable to express information at different levels of granularity adapted to the user. These answers reflect the insight in the technical evaluation that *Text* enables the representation of ISC and additional information in a compact way.

The interview prototype presents processes, instances, ISC, and ISC instances side by side. We asked the stakeholders if this *Presentation* meets their expectations. One stakeholder opted for separation of concerns, i.e., process/ISC on one side and instances on the other. Another stakeholder prefers the ISC instances as optional information. The third stakeholder stated that for some use cases a pure process and ISC visualization is sufficient, for other use cases all instances are necessary. Overall the stakeholders mentioned that the type of presentation depends on the role and use case. Based on these answers a flexible system that allows to hide and rearrange parts of the visualization is a good choice.

Stakeholders were divided on whether desktop, *AR*, or *VR* is preferred. One stakeholder was more skeptical in terms of *AR/VR* and is personally comfortable with desktop. However, the stakeholder can imagine that *AR/VR* brings advantages when visualizing process models and their ISC. The others prefer *AR/VR* over desktop and can imagine various applications, e.g., *AR* could be used on the shopfloor to display the process instances directly on the machines.

Limitations and Threats to Validity: We counter threats to validity with several measures. Firstly, the visualizations are based on various works from literature. Secondly, research bias is addressed as questions asked during the interview were defined in advance. Lastly, the stakeholders were not involved

in the development/research in any kind. For the evaluation, we used process models from two domains, i.e., logistics and manufacturing. Since we have used ISC from each category of the ISC classification for the the logistics domain, we think that the presented results are transferable to other domains.

6 Conclusion and Outlook

This work evaluates compliance visualization approaches from literature with respect to complex process scenarios with instance-spanning constraints (ISC). The evaluation is based on literature, a technical evaluation, and in-depth interviews with stakeholders. In summary, for assessing compliance states, we recommend *Color* for *gaining a quick overview* and *Text* for *in-detail analysis*. Stakeholders favor a visual indicator showing that a rule is violated, i.e., *Size*, *Rotation*, and *Shape*. In future work we want to investigate the combination of multiple visualization approaches as suggested by the stakeholders. Further directions include the investigation of quantitative and qualitative requirements such as contradictions, subsumption, and root-cause.

Acknowledgments

This work has been partially supported and funded by the Austrian Research Promotion Agency (FFG) via the *"Austrian Competence Center for Digital Production"* (CDP) under the contract number 881843.

References

1. Awad, A., Weske, M.: Visualization of compliance violation in business process models. In: Business Process Management Workshops. pp. 182–193 (09 2009)
2. Becker, J., Delfmann, P., Dietrich, H., Steinhorst, M., Eggert, M.: Business process compliance checking - applying and evaluating a generic pattern matching approach for conceptual models in the financial sector. Inf. Syst. Front. 18(2), 359–405 (2016)
3. Bertin, J.: Semiology of graphics; diagrams networks maps. Tech. rep. (1983)
4. Böhmer, K., Rinderle-Ma, S.: Mining association rules for anomaly detection in dynamic process runtime behavior and explaining the root cause to users. Inf. Syst. 90, 101438 (2020)
5. Boyce, C., Neale, P.: Conducting in-depth interviews: a guide for designing and conducting in-depth interviews for evaluation input. (2006)
6. David, A., Larsen, K.G., Legay, A., Mikučionis, M., Poulsen, D.B.: Uppaal smc tutorial. Software Tools for Technology Transfer 17(4), 397–415 (2015)
7. Fdhila, W., Gall, M., Rinderle-Ma, S., Mangler, J., Indiono, C.: Classification and formalization of instance-spanning constraints in process-driven applications. In: Business Process Management. pp. 348–364 (2016)
8. Gall, M., Rinderle-Ma, S.: Visual modeling of instance-spanning constraints in process-aware information systems. In: Advanced Information Systems Engineering. pp. 597–611 (2017)

9. Gall, M., Rinderle-Ma, S.: From instance spanning models to instance spanning rules. In: Enterprise, Business-Process and Information Systems Modeling. pp. 131–146 (2018)
10. Gall, M., Rinderle-Ma, S.: Assessing process attribute visualization and interaction approaches based on a controlled experiment. Int. J. Cooperative Inf. Syst. 29(4), 2050007:1–2050007:33 (2020)
11. Gonçalves, L., Leta, F.: Macroscopic rock texture image classification using a hierarchical neuro-fuzzy class method. Math. Problems in Eng. 2010 (06 2010)
12. Koetter, F., Kintz, M., Kochanowski, M., Wiriyarattanakul, T., Fehling, C., Gildein, P., Wagner, S., Leymann, F., Weisbecker, A.: An universal approach for compliance management using compliance descriptors. In: Cloud Computing and Services Science. pp. 209–231 (2017)
13. Li, H., Yu-Shun, F.: Workflow model analysis based on time constraint petri nets. Journal of Software 15 (01 2004)
14. Louise Barriball, K., While, A.: Collecting data using a semi-structured interview: a discussion paper. Journal of advanced nursing 19(2), 328–335 (1994)
15. Ly, L.T., Maggi, F.M., Montali, M., Rinderle-Ma, S., van der Aalst, W.M.P.: Compliance monitoring in business processes: Functionalities, application, and tool-support. Inf. Syst. 54, 209–234 (2015)
16. Ly, L.T., Rinderle-Ma, S., Knuplesch, D., Dadam, P.: Monitoring business process compliance using compliance rule graphs. In: On the Move to Meaningful Internet Systems. pp. 82–99 (2011)
17. Maggi, F.M., Montali, M., van der Aalst, W.M.P.: An operational decision support framework for monitoring business constraints. In: Fundamental Approaches to Software Engineering. pp. 146–162 (2012)
18. Montali, M., Maggi, F., Chesani, F., Mello, P., Aalst, W.: Monitoring business constraints with the event calculus. ACM Transactions on Intelligent Systems and Technology 5 (12 2013)
19. Moody, D.L.: The "physics" of notations: Toward a scientific basis for constructing visual notations in software engineering. IEEE Transactions on Software Engineering 35(6), 756–779 (2009)
20. Reichert, M., Bassil, S., Bobrik, R., Bauer, T.: The proviado access control model for business process monitoring components. Enterprise Modelling and Information Systems Architectures 5(3), 64–88 (2010)
21. Rinderle-Ma, S., Gall, M., Fdhila, W., Mangler, J., Indiono, C.: Collecting examples for instance-spanning constraints. arXiv preprint arXiv:1603.01523 (2016)
22. Rinderle-Ma, S., Kabicher-Fuchs, S.: An indexing technique for compliance checking and maintenance in large process and rule repositories. Enterp. Model. Inf. Syst. Archit. Int. J. Concept. Model. 11, 2:1–2:24 (2016)
23. Saraiya, P., Lee, P., North, C.: Visualization of graphs with associated timeseries data. In: IEEE Symposium on Information Visualization. pp. 225–232 (2005)
24. STAMFORD Conn.: Gartner predicts legal technology budgets will increase threefold by 2025 (2021), `https://gtnr.it/3qP2ySd`
25. Vierhauser, M., Rabiser, R., Grünbacher, P., Egyed, A.: Developing a dsl-based approach for event-based monitoring of systems of systems: Experiences and lessons learned (e). In: Automated Software Engineering. pp. 715–725 (2015)
26. Wieringa, R.J.: Design Science Methodology for Information Systems and Software Engineering. Springer (2015)