A Visual Narrative Path From Switching to Resuming an RE Task

Zahra Shakeri Hossein Abad, Alex Shymka, Jenny Le, Noor Hammad, Guenther Ruhe
Department of Computer Science
University of Calgary, Calgary, Canada
Email: {zshakeri, alex.shymka, Jenny.le, nour.hammad, ruhe}@ucalgary.ca

Abstract—Requirements Engineering (RE) is closely tied to other development activities and is at the heart and foundation of every software development process. This makes RE the most data and communication intensive activity compared to other development tasks. The highly demanding communication makes task switching and interruptions inevitable in RE activities. While task switching often allows us to perform tasks effectively, it imposes a cognitive load and can be detrimental to the primary task, particularly in complex tasks as the ones typical for RE activities. Visualization mechanisms enhanced with analytical methods and interaction techniques help software developers obtain a better cognitive understanding of the complexity of RE decisions, leading to timelier and higher quality decisions. In this paper, we propose to apply interactive visual analytics techniques for managing requirements decisions from various perspectives, including stakeholders communication, RE task switching, and interruptions. We propose a new layered visualization framework that supports the analytical reasoning process of task switching. This framework consists of both data analysis and visualization layers. The visual layers offer interactive knowledge visualization components for managing task interruption decisions at different stages of an interruption (i.e. before, during, and after). The analytical layers provide narrative knowledge about the consequences of task switching decisions and help requirements engineers to recall their reasoning process and decisions upon resuming a task. Moreover, we surveyed 53 software developers to test our visual prototype and to explore more required features for the visual and analytical layers of our framework.

I. INTRODUCTION AND MOTIVATION

Task interruptions are a type of task switching or sequential multitasking [1]. During the term of performing a task, professional software developers need to frequently interrupt their unfinished tasks for various reasons, such as answering unexpected questions from their coworkers, to address new re-prioritized requirements, or to attend a scheduled team meeting. According to the results of the retrospective analysis we conducted on 5,079 recorded tasks of 19 professional software developers (submitted to RE'17 [2]), we found that in 98% of cases that the disruptiveness of Requirements Engineering (RE) interruptions is statistically different from other software development tasks, RE related tasks are more vulnerable to interruptions compared to other task types. This is mainly due to the high level of cognition in RE activities. As illustrated in Figure 1, to execute a typical software development task, the subject might take the direct route to proceed from the initial state of a task (q_0) to the final state (q_f). Using the direct link, without any detours on the way, is only one possible way
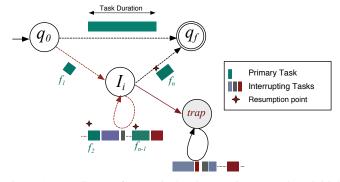

Fig. 1: A state diagram for a typical task execution [q_0 and q_f: initial and final states, f= task fragments, and I_i= the ith interruption]

of doing it. However, this is an unrealistic view of the way software developers perform their tasks. The execution of task might be interrupted by others tasks. Node I_i represents these interruptions and the interruption loop on this node shows the nested interruptions that might occur during the term of executing the primary task. Considering the limited cognitive flexibility of humans [3], nested task switchings causes mental congestion for keeping track of multiple states of tasks, which decays the goal of the primary task [4]. Thus, the greater the value of i, the more disruptive the interruptions. In our recent study on exploring and understanding tasks interruption in RE activities [2], we found that for 66% of RE interruptions users did not resume the task right away, and 11% of interrupted RE tasks never got resumed (i.e. the trap state on Figure 1).

In recent years, much research effort has been directed towards visualizing different aspects (e.g. activities, artifacts) of RE [5]–[7]. However, to the best of our knowledge, there is no previous work on the application of visual analytics approach for reducing the cognitive cost of interruptions and task switching in RE activities. This paper reports on a proposed visual analytics framework, which provides a visual narrative solution for managing interruption decisions and supports the analytical reasoning process of primary task resumption. This layered framework consists of three analytical and three visual layers and all of these layers are interrelated. The analytical layers aim to analyze historical interrupted tasks and their associated artifacts and provide additional insights into the visual layers. Moreover, these layers will be used to analyze the interactions histories in terms of various RE artifacts and provide an analytical reasoning process for RE task resumptions. The visual layers offer interactive visualization which covers the

main steps (i.e. before, during, after) of an interruption. We provide background information about our study in Section II, followed by our research goals and research questions (Section III). Our proposed visual framework, including our overall research approach and our progress on each layer of this framework, is discussed in Section IV. We conclude the paper by reporting the main expected contributions and our evaluation plan in Section V.

II. BACKGROUND AND RELATED WORK

A. Terminologies (Task Interruptions in RE)

In this section, we briefly summarize the main terminologies that we use in this paper.

Definition 1: The *disruptiveness* of an interruption refers to the negative impact they pose on developers' productivity and can be measured in terms of the number of task fragments resulted from this interruption (D_1), and the length of resumption (D_2) and interruption (D_3) lags [4], [8].

Definition 2: We identified a set of *interruption characteristics* in [2], which will be used as the key data points of our visualization methods. A task's context (i.e. project), type, granularity level, progress status, and priority, as well as the timing of interruptions are some example of these features.

Definition 3: In our recent study, we found that in the context of RE task switching and interruption, *self-interruptions* (i.e. interruptions initiated by the subject) are more disruptive. In this paper, we use this concept to design one of the visual components of our proposed visual framework.

B. RE Visualization Techniques

A classification of existing RE visualization techniques based on the visualization type they address (e.g. data, information, or knowledge visualization) and the aspects of RE they cover (e.g. RE activities, Stakeholders, and domain) is presented in our recent Systematic Literature Review (SLR) of RE visualization [7] (presented in RE'16).

In this SLR we proposed the common RE visualization patterns in the form of ⟨**content**, **focus**⟩, where **content** shows each of the RE activities, and **focus** denotes the *What, How, Why, Where,* and *Who* components of each visualization technique. The results of this SLR revealed that there is a clear need for more investigation and research to support knowledge visualization in the area of RE. Moreover, among all RE activities, requirements communication and evolution, as well as Non-Functional Requirements (NFR's) and requirements uncertainties need more investigation.

C. Interruption Visualization

Parnin et al. [8] explore the various strategies and coping mechanisms that programmers utilize in order to manage interruptions, while also proposing suggestions on how task resumption can be better supported. A system for handling interruptions is described as having three phases; suspension, resolution and resumption. During the suspension phase, the programmer has the choice to preserve their working state by internalization (i.e use memory training to remember the working state) and externalization (i.e use physical or electronic tools to preserve the working state). Internalization and externalization can be utilized in three suspension strategies: rehearsal, serialization, and cue priming. In the resolution phase, the programmer uses various tactics to restore their previous working state. Some restoration techniques are: global restoration, goal restoration, and plan restoration. Lastly, resumption is the last step the programmer would go through before resuming their primary task. Some popular resumption tactics are: return to last stopping place; review task assignment; execute program; restore from task breakdown; and review source code change history. Different programmers would perform the resumption strategy that they are more familiar with.

Liu et al. [9] conducted an experiment where visual feedback was meant to motivate users to return to the primary task. The visual feedback was either a progress bar that depleted or a flower that withered as the user strayed from the primary task. The task was editing a Microsoft Word document for 40 minutes with no end state. They encouraged participants to go about this task at their leisure, allowing any habits they had such as listening to music or checking emails. Participants with visual feedback spent less time on other tabs and more time editing the word document than the control group with no visual feedback.

While the existing research provided a wealth of insight on RE visualization and resumption strategies, to the best of our knowledge, there has been no research in the RE community that investigates a visual aid for reducing the cognitive cost of RE task switchings and interruptions and to help requirements engineers' recall and reconstruct their reasoning process.

III. GOALS AND RESEARCH QUESTIONS

We intend to develop a narrative visual framework to aid requirements engineers in making swift decisions regarding their task switchings without dealing with complex visualization methods or, worse yet, to deal with the data directly. To this end, we raised the following research questions:

- **RQ1 [Before Interruption]:** What are effective visualization techniques for making an efficient RE task switching decision (the $q_0 \rightarrow I_i$ transition, Figure 1)?
- **RQ2 [Suspension Period]:** What are visualization techniques to monitor the number of task fragments resulted from interruptions (the self-loop on state I_i in Figure 1)?
- **RQ3 [After Interruption]:** What are effective cues and visualization techniques for resuming tasks with less cognitive cost (all sections marked with ✦ on Figure 1)?

IV. A LAYERED VISUALIZATION FRAMEWORK

An overview of our research approach is modelled in a form of a layered visualization framework and is presented in Figure 2. In this section, we present the details of our research approach and our current progress on each layer of this framework in detail.
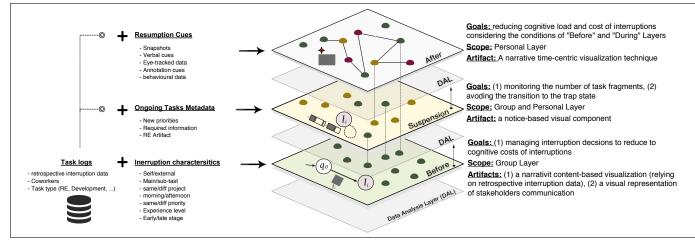
Fig. 2: The Proposed Interruption Visualization Framework

A. Data Analysis Layers

In this section we describe the overall data analysis associated with each *"Data Analysis Layer (DAL)"* of our proposed framework.

1) Mining Interruption Patterns: To visualize the **before interruption** and **suspension** layers, we apply and customize the Apriori algorithm [10] to mine association rules and explore *disruptiveness patterns*. We define a *disruptiveness Pattern* as $D = \langle (T_k, \alpha_i), D_{1-j} \rangle$, where T_k represents the type of an RE task (e.g. requirements gathering, analysis, evolution, or validation); α_i denotes interruption characteristics, such as contextual, temporal, and type of the interruptions; and D_j represents the disruptiveness factors (Section II-A), such as interruption lag, resumption lag and the number of fragments resulted from each RE interruption.

An *association rule* $(Z \Rightarrow Y)$ for a pattern $D = \langle (T_k, \alpha_i), D_j \rangle$ consist of two non-empty sets:

$$Z = \bigcup_{k=1}^{\#types} \bigcup_{j=1}^{n} (T_k, \alpha_i) \quad \text{and} \quad Y = \{D_1, D_2, D_3\}$$

and is interpreted as: "whenever an RE activity k gets interrupted with interruption characteristics $\{\alpha_1, \alpha_2, ..., \alpha_n\}$, we have specific disruptiveness measures $\{D_1, D_2, D_3\}$". The *support* of a *disruptiveness pattern* is the percentage of records which contain all parameters in $Z \cup Y$ and can be used as:

$$support = \frac{freq(Z, Y)}{|\,Itemset\,|}$$

The *confidence* of an association rule $(Y \Rightarrow Z)$ indicates the strength of the rule and can be determined as conditional probablity $P(Z \mid Y)$ that the disruptiveness value Z occurred, given the interruption condition Y already happened. According to the Apriori algorithm, a pattern occurs frequently if its support is above the $\min_{support}$, and an association rule is strong if its confidence is above the $\min_{confidence}$ values.

An illustrative case study: We recorded the task switchings and interruptions of a real world on-going software development project. For the sake of simplicity, we only chose five random "requirements modeling" tasks and ran our case study with only three interruption characteristics.

The task-characteristics matrix, as the main input of the process, is illustrated in Figure 3. The Apriori algorithm takes a $Min_{support}$ and a $Min_{confidence}$ as input parameters. To run this illustrative example, we used the following values: $\begin{cases} Min_{support} & 0.5 \\ Min_{confidence} & 0.5 \end{cases}$ for these parameters.

For a pattern $D = \langle (T_k, \alpha_i), D_j \rangle$ there exist $2^{|T_k \cup \alpha_i \cup D_j|} - 1$ association rules. To deal with the complexity of our frequent pattern mining approach, we filtered our item sets as follows:

- We only consider one RE task type for running the algorithm (the type of the on-going task). For example, if the user intends to switch a requirement modeling task, we only consider this type of task for exploring frequent patterns, as illustrated in Figure 3.
- We ignore all item sets which only belong to one of the *interruption characteristics* or *disruptiveness measures* sets. For example, as illustrated in Figure 3, iteration 2, we do not have any item set like $\{D_{1high}, D_{3high}\}$.

The algorithm iterates over the set of item sets (i.e. the input box on Figure 3) and forms patterns from the interruption characteristics and disruptiveness measures in the same set. We expanded patterns in each iteration based on the *support* parameter. The Iterative process continues until all possible extensions are reached (e.g. the 3rd iteration in this example). The final set of frequent patterns of our study contains only one frequent pattern: *"Self-switching a requirements modeling task in the morning contributes to a greater interruption lag"*, with confidence of 100% and support 66%. The frequent patterns resulted from this layer will be used as an input for the first and the second narrative visualization layers (Figure 2).

2) Analysis of Interaction Histories: The design decisions for the third layer (i.e. the resumption layer) are impacted by analyzing the data collected from users interaction histories. To model these histories we use a directional graph of resumption states. Each resumption strategy (e.g. verbal cues, eye-movement data, thumbnail images) is represented as a single node and edges model the sequence of interactions with each of these cues. To discover time-ordered sequences of cues that have been followed by past users, who have been in the same situation, we use the Sequen-
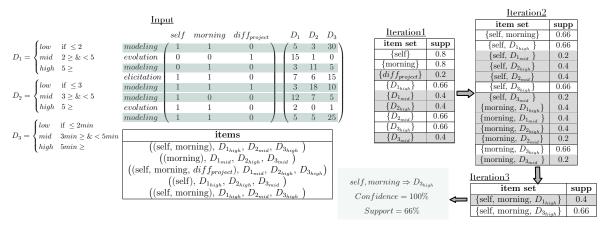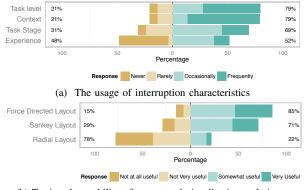
Input

$$D_1 = \begin{cases} low & if \leq 2 \\ mid & 2 \geq \& < 5 \\ high & 5 \geq \end{cases}$$

$$D_2 = \begin{cases} low & if \leq 3 \\ mid & 3 \geq \& < 5 \\ high & 5 \geq \end{cases}$$

$$D_3 = \begin{cases} low & if \leq 2min \\ mid & 3min \geq \& < 5min \\ high & 5min \geq \end{cases}$$

	self	morning	$diff_{project}$	D_1	D_2	D_3
modeling	1	1	0	5	3	30
evolution	0	0	1	15	1	0
modeling	0	1	0	3	11	5
elicitation	1	1	0	7	6	15
modeling	1	1	1	3	18	10
modeling	1	0	0	12	7	5
evolution	1	1	0	2	0	1
modeling	1	1	0	5	5	25

items

$((self, morning), D_{1_{high}}, D_{2_{mid}}, D_{3_{high}})$
$((morning), D_{1_{mid}}, D_{2_{high}}, D_{3_{mid}})$
$((self, morning, diff_{project}), D_{1_{mid}}, D_{2_{high}}, D_{3_{high}})$
$((self), D_{1_{high}}, D_{2_{high}}, D_{3_{mid}})$
$((self, morning), D_{1_{high}}, D_{2_{mid}}, D_{3_{high}})$

Iteration1

item set	supp
{self}	0.8
{morning}	0.8
{$diff_{project}$}	0.2
{$D_{1_{high}}$}	0.66
{$D_{1_{mid}}$}	0.4
{$D_{2_{high}}$}	0.4
{$D_{2_{mid}}$}	0.66
{$D_{3_{high}}$}	0.66
{$D_{3_{mid}}$}	0.4

Iteration2

item set	supp
{self, morning}	0.66
{self, $D_{1_{high}}$}	0.66
{self, $D_{1_{mid}}$}	0.2
{self, $D_{2_{high}}$}	0.4
{self, $D_{2_{mid}}$}	0.4
{self, $D_{3_{high}}$}	0.66
{self, $D_{3_{mid}}$}	0.2
{morning, $D_{1_{high}}$}	0.4
{morning, $D_{1_{mid}}$}	0.4
{morning, $D_{2_{high}}$}	0.4
{morning, $D_{2_{mid}}$}	0.2
{morning, $D_{3_{high}}$}	0.66
{morning, $D_{3_{mid}}$}	0.2

Iteration3

item set	supp
{self, morning, $D_{1_{high}}$}	0.4
{self, morning, $D_{3_{high}}$}	0.66

$self, morning \Rightarrow D_{3_{high}}$
$Confidence = 100\%$
$Support = 66\%$

Fig. 3: An illustrative case study of the application of association rule mining for layers 1 and 2

(a) The usage of interruption characteristics

(b) Testing the usability of prototyped visualization techniques

Fig. 4: Survey results for required data points on the graphs and usability of the communication graphs

tial Association Mining (SAM) approach, as detailed in [11]. We use the following format to define each item set: $\langle (cue_1, t_{1,1}), (cue_2, t_{2,1}), ..., (cue_m, t_{i,k}) \rangle$, where i represents the order of using cues and k represents the number of times a user navigates to a cue. For example, a possible item set for the sample graph presented in Figure 5 (a) can be defined as: $\langle (Eye, t_{1,1}), (Verb, t_{2,1}), (Eye, t_{3,2}) \rangle$. Each SAM has a degree of support and confidence associated with it. The support of a rule is defined as the fraction of strings in the set of interaction strings, where the rule successfully applies. We use these patterns to visualize the third layer of our visualization framework.

B. Visualization Layers

The visual layers of the proposed framework (Figure 2) consist of visual narrative representations of the main transitions of interruption graph, illustrated in Figure 1: (1) Layer 1 (before interruption) models the $q_0 \rightarrow I_i$ transition, (2) Layer 2 (suspension Layer) provides a visual analytics component to monitor the self-loop on state I_i, and (3) Layer 3 (after interruption) addresses the resumption times, annotated by ◆, and aims to reduce the cognitive cost of interruptions.

User Survey: To gain more insight into the required feature for each layer and to test the usability of our developed prototype, we surveyed 53 software developers. We used Survey Monkey[1] to design the online survey and collect responses. To

recruit our survey participants, we used snowball and random sampling (e.g. LinkedIn, Reddit Forums) methods [12]. The survey consists of 10 questions including multiple choice, Likert scale, and open-ended questions. In addition to these questions, we included a direct link to the developed prototype in our survey and asked our participants to interact with each visualization technique and rate them based on their usability and usefulness in representing stakeholders' communication [2]. The average software development experience of participants was 4.5 (± 3), and the average of their teamwork experience was 3.5 (± 3) years. This study has been approved by the University of Calgary Conjoint Faculties Research Ethics Board (CFREB [3]).

1) Layer 1- Before Interruption Visualization: We refer to this layer, as detailed in Figure 2, as a group layer, which expresses the collaborative aspects of task switchings. This layer consists of the fundamental dimensions of a visualization; *data, information*, and *knowledge* visualization as follows:

- Data Visualization: This dimension covers graphical representation of unprocessed information including; (1) all influential factors on the disruptiveness of interruptions, such as type, priority, stage, context (i.e. project) and level of all on-going tasks, as well the experience level of task performers. The majority of our survey participants also use these parameters before switching their tasks (Figure 4 (a)); (2) main triggers of self-interruptions such as boredom and task blockage. To explore the data points of the second category, in addition to our main survey, we asked an online open-ended question from 23 professional software developers about the main reasons of switching their RE tasks. 7 (30%) stated that they often switch their RE tasks when they get blocked and cannot proceed with the primary task. Lack of information about the current requirements, waiting for stakeholders' feedback, and conflicting requirements are the common reasons for task blockage, as stated by our participants. 6 (26%) described the "re-prioritization" of requirements as the main reason of their self-interruptions. "Getting bored" and "personal schedules" come next, with each of them being 5 (22%) and 4 (17%) participants. One participant also stated

---

[1]http://www.surveymonkey.com

[2]https://wcm.ucalgary.ca/zshakeri/files/zshakeri/visualization-survey.pdf
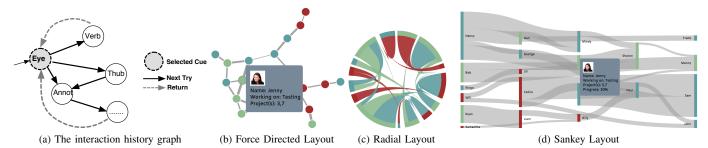[3]http://www.ucalgary.ca/research/researchers/ethics-compliance/cfreb

Fig. 5: Analytical and visual components of different layers of the proposed visual framework

that *"I usually get distracted by researching a new technology that could help solve the task at hand"*. While the data points related to disruptiveness factors cover the "re-prioritization" and "personal schedules", we still needed to add boredom and task blockage to this layer to minimize the number of self-interruptions.

To be able to create distinct visual representations for these data points, we proposed to use *visual variables*, such as size, shape, orientation, color, value, and texture [13]. However, choosing an appropriate visual variable to represent each aspect of the data is not a straightforward process. A small change in a particular visual variable can significantly affect the performance of a particular task. To represent the *boredom* and *requirements uncertainties and conflicts* (i.e. task blockage), we used the hue, fuzziness, texture, and objects, as the main visual variables and asked our survey participants to choose the appropriate variable for each of these data points. 30 (59%) participants chose the *color* (i.e. red) variable and 28 (54%) chose *objects* as the most appropriate variables for representing requirements uncertainty and employee's boredom, respectively.

- Information Visualization: To add more insight to the data points listed above, we added the stakeholders' communication graph, which represents the task switching frequencies based on tasks' subjects. To this end, we prototyped the three main visualization techniques for representing communication, such as Force Directed Layout (Figure 5(b)), Radial Layout (Figure 5(c)), and Sankey Layout (Figure 5(d)). The nodes on each layout are people in a communication network, the thicker the line between nodes, the higher the volume of task switching requests. To evaluate the usability of these techniques, we asked our participants to interact with each visualization and rate them based on their usefulness in understanding the communications. As illustrated in Figure 4(b), 45 (85%) participants chose the "Forced Directed Layout" (Figure 5 (b)) as a useful technique for understanding the state of the other on-going tasks and the behavior of the communication among stakeholders. For example, a participant stated: *"this technique clearly shows the relationships between each node; good labels when hovering over; good overall representation and different colors of the relationships"*. We also received some feedback for improving this prototype for the next release, as in: *"I see connections between members but I wasn't able to decipher exactly how they are connected or by what projects. Perhaps pre-loading the profile cards rather*

than activating only when hovered over". Moreover, 38 (71%) participants found the "Sankey Layout" (Figure 5 (c)) technique useful, as one participant stated: *" this technique lets me know quickly who I was communicating with"*. However, some participants found this technique confusing and hard to figure out the flow of information effectively. For the next iterations of improving our visual prototype, we will exclude the "Radial Layout" technique (Figure 5 (d)) from our proposed visual framework, as 41 (78%) participants did not find this approach useful, as in *"no clue what I am looking at"*. However, this technique might be useful for representing the tracking of resumption cues (Layer 3), which needs more investigation in future studies.

- Knowledge Visualization: The output of the first analytical layer (i.e. the disruptiveness and interaction patterns), explained in Section IV-A, are used to provide a visual narrative knowledge in this layer.

2) Layer 2- Suspension Period: This layer aims to (1) monitor the number of task fragments resulted from each interruption; (2) shorten the resumption lag; and (3) to avoid the trap situation (i.e. the situation where the interruptee never return to the interrupted task). We asked our participant whether they use any tool or technique to remind them to return to the primary task, and how they would design this reminder. 37 (69%) participants stated they do not use any technique or tool for managing the resumption lag of their interrupted task. The participants predominantly described the main features of this reminder as follows:

1) **Notifications [43(81%)]:** Pop-ups, verbal notifications, an encouraging email, and sound effects.
2) **Visual pins [10 (19%)]:** Representing an image of the interrupted task on the screen, and open tabs on an IDE.

Some participants provided more details about the timing and main features of the notifications and visual pins, as in: *"It should appear when I'm between tasks, or at least at a reasonable pausing point, which might be indicated when I type "git commit" or "git push"* and in: *"It would know the priority of my tasks and remind me at the most convenient time with my schedule."*.

Moreover, the disruptiveness patterns produced by applying association rule mining (i.e. Apriori algorithm), as described in Section IV-A, will be used in this layer to add a narration and insight to the visual reminder.

3) Layer 3- Resumption Time: This layer aims to provide a visual narrative of what users need to reconstruct their

TABLE I: Participants' responses to the usefulness of various resumption cues (1= most useful, 5= least useful)

Cues	1	2	3	4	5
Annotation cues	46%	17%	2%	21%	12%
Thumbnail images	15%	37%	26%	12%	10%
Verbal cues	24%	12%	17%	24%	22%
Eye cues	12%	20%	22%	24%	22%
Behavior graph	5%	15%	29%	17%	34%

memory after resuming a task. Our survey participants reported that, on average, they need 3.2 $(-3, +12)$ minutes to refresh their memory about what they were doing before getting interrupted. Visualizations of interaction logs, as stated by Lipford et al. [14], can serve as an effective memory aid, allowing analysts to recall additional details of their strategies and decisions. Capturing low-level user actions such as mouse events, eye tracking [15], and keyboard events [16], as well as retrospective verbalization [14], and annotating the task artifacts [16] are common approaches to cue the resumption process. We asked our survey participants to rank the usefulness of cues, listed in Table I, for constructing their memory state after resuming a task. 33 (63%) and 28 (52%) participants found "annotation cues" and "thumbnail images" more useful technique for recalling their reasoning process, compared to other techniques.

Moreover, the *interaction history patterns* produced by applying the sequential association mining technique, will be used at this layer to design the order and the navigation of resumption cues. In addition, the type of an RE task will be considered in producing the interaction patters. For instance, the appropriate resumption cue for a requirements modeling task might not be such useful for resuming a requirements specification task. As the artifacts of these two tasks are different and have different levels of complexity.

V. EXPECTED CONTRIBUTIONS AND EVALUATION

The main contribution of this paper lies primary in providing a multi-layered hybrid visualization technique for better managing requirements communication and interruptions. In summary, this research aims at: (1) exploring a novel hybrid visualization technique to reduce the cognitive load of requirements communication and interruptions; (2) providing a new narrative "knowledge visualization" technique (i.e. storytelling) in the area of RE by integrating data analysis and visualization techniques; and (3) proposing a set of time-centric visualization techniques for reducing the cognitive cost of RE task interruptions and classify these techniques based on RE activities and artifacts (e.g. requirements elicitation, communication, and evolution). We addressed our current progress on RQ1-3 in Section IV. Further, we presented an illustrative case study with different interruption characteristics (e.g. self/external, time, and task context) and disruptiveness measures (D_{1-3}) to clarify our approach for implementing the analytical layers.

Regarding the evaluation of our proposed approach, we plan to design and run several user studies to empirically evaluate the effectiveness of our approach and the usability of proposed visualization technique. To assess the performance of the analytical approach we use in analytical layers (i.e.

association rule mining and sequential association mining), we plan to run our framework on different scales of real world datasets. However, the filtering method we described in Section IV-A helps reduce the complexity of large-scale datasets. In summary, we believe that this research will foster a clear connection between various areas of research: requirements engineering, data analysis, data visualization, and Human Computer Interaction (HCI).

VI. ACKNOWLEDGEMENT

REFERENCES

[1] D. D. Salvucci and N. A. Taatgen, *The multitasking mind*. Oxford University Press, 2010.

[2] Z. Shakeri Hossein Abad, G. Ruhe, and M. Bauer, "Task interruptions in requirements engineering: reality versus perceptions!" In *the 25th IEEE International Conference on Requirements Engineering*, 2017 under review.

[3] B. Vasilescu, K. Blincoe, Q. Xuan, C. Casalnuovo, D. Damian, P. Devanbu, and V. Filkov, "The Sky is Not the Limit: Multitasking Across GitHub Projects," in *Proceedings of the 38th International Conference on Software Engineering*, ACM, 2016, pp. 994–1005.

[4] G. Mark, V. M. Gonzalez, and J. Harris, "No task left behind?: examining the nature of fragmented work," in *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, ser. CHI '05, ACM, 2005, pp. 321–330.

[5] G. Lucassen, F. Dalpiaz, J. M. E. van der Werf, and S. Brinkkemper, "Visualizing user story requirements at multiple granularity levels via semantic relatedness," in *Conceptual Modeling: 35th International Conference, ER 2016, Gifu, Japan, November 14-17, 2016, Proceedings 35*, Springer, 2016, pp. 463–478.

[6] D. v. d. Linden, I. Hadar, and A. Zamansky, "Towards a marketplace of visual elements for notation design," in *2016 IEEE 24th International Requirements Engineering Conference (RE)*, 2016, pp. 353–358.

[7] Z. S. H. Abad, M. Noaeen, and G. Ruhe, "Requirements engineering visualization: a systematic literature review," in *2016 IEEE 24th International Requirements Engineering Conference (RE)*, 2016, pp. 6–15.

[8] C. Parnin and S. Rugaber, "Resumption strategies for interrupted programming tasks," *Software Quality Journal*, vol. 19, no. 1, pp. 5–34, 2011.

[9] Y. Liu, Y. Jia, W. Pan, and M. S. Pfaff, "Supporting task resumption using visual feedback," in *Proceedings of the 17th ACM conference on Computer supported cooperative work & social computing*, ACM, 2014, pp. 767–777.

[10] R. Agrawal, R. Srikant, *et al.*, "Fast algorithms for mining association rules," in *Proc. 20th int. conf. very large data bases, VLDB*, vol. 1215, 1994, pp. 487–499.

[11] E. Frias-Martinez and V. Karamcheti, "A prediction model for user access sequences," in *WEBKDD Workshop: Web Mining for Usage Patterns and User Profiles*, 2002.

[12] F. Shull, J. Singer, and D. I. Sjøberg, *Guide to advanced empirical software engineering*. Springer, 2008, vol. 93.

[13] M. Carpendale, "Considering visual variables as a basis for information visualisation," 2003.

[14] H. R. Lipford, F. Stukes, W. Dou, M. E. Hawkins, and R. Chang, "Helping users recall their reasoning process," in *2010 IEEE Symposium on Visual Analytics Science and Technology*, 2010, pp. 187–194.

[15] N. Eger, L. J. Ball, R. Stevens, and J. Dodd, "Cueing retrospective verbal reports in usability testing through eye-movement replay," in *Proceedings of the 21st British HCI Group Annual Conference on People and Computers: HCI...But Not As We Know It - Volume 1*, ser. BCS-HCI '07, British Computer Society, 2007, pp. 129–137.

[16] Y. B. Shrinivasan and J. J. van Wijk, "Supporting the analytical reasoning process in information visualization," in *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, ser. CHI '08, ACM, 2008, pp. 1237–1246.