**ReVA: Visual Analytics Interaction log pattern Visualization   
supporting Recover and Reuse**

Youngtaek Kim(김영택), Yieun Kim (김이은)

[ytaek.kim@hcil.snu.ac.kr](mailto:ytaek.kim@hcil.snu.ac.kr), [yekim@hcil.snu.ac.kr](mailto:yekim@hcil.snu.ac.kr), [yuhoon.ki@samsung.com](mailto:yuhoon.ki@samsung.com)

Abstract

Visual Analytics(VA) allows people to analyze data when the don't know exactly what questions they need to ask in advance. In case of ell-defined questions, purely computational techniques and proper algorithm will be enough. However, if the problem is ill specified, they explore raw data in various and repetitive way until good questions and answers are found. VA tools such as excel, Tableau, spotfire and so on help them to work effectively, however, during exploring process, repetitive and tedious works occurred. For decreasing duplicate jobs, most VA tools provide several functions such as saving current status or templates. But as far as we know, these functions are limited to mainly visual encoding idioms without interactions. In this proposal, we introduce interactions tracking and reproducing techniques.

**Keywords**: interaction log, visualization, recover and reuse

**Index Terms**: Visual Analytics,

# Introduction

Sharing and reusing each person’s insight and reasoning process obtained from Visual Analytics (VA) is regarded quite difficult job. Clear and well-defined problems can be solved by purely computational techniques and sharing the analyzed results is also possible. However, in case of ill-defined problems or not knowing what questions required to be asked in advance, enabling reasoning process during Visual Analytics to be recovered and reused is necessary. Existing VA Tools such as Excel, Spotfire, Tableau and so on provide history functions but it is just recall of linear history of manipulating the tools. In this proposal, our contributions are as below. - Define how to descript practical and reusable visual encoding/interaction logging pattern - Visualize captured patterns and provide recover & replace interfaces with a prototype application.

Our demands on this project comes from software-engineering domain. Recently various and tremendous data during software development is being produced via heterogeneous system and tools: SCM(Source Code Management), ITS(Issue Tracking System), mailing list, Static Analysis Tools, code review systems and so on. However, developers or software managers tend not to be accustomed to handle these wide data and depend on relatively basic analysis view which is default-embedded in system. Obtaining key problems and sharing reasoning process from data is necessary in this domain.

# Related Works

[1] introduces five interrelated states in the research of visual analytics provenance: perceive, capture, encode, recover and reuse. In “Perceive” stage, the understanding of how information is perceived by the user. “Capture” stage means tracking and logging not only literally a user’s linear sequence of actions but related semantic information. “Encode” refers to describing the captured provenance in predefined formats. Captured and encoded the user’s provenance enables to be “recovered” in the same manner and also “reused” by reapplying it to a new data or domain. In the implementation of this project, we treat mainly “recover” and “reuse” stage due to the lack of time and resource. Regarding “Capture” and “Encode”, we will suppose that users’ interaction logging data exists and then manipulate it manually. As the result, the visualization and interactions of recovering and reusing stage will be developed as a rapid prototype of this project

.

GraphTrail [2], which is an interactive visualization for analyzing multivariate and heterogeneous networks, provides functions that capture users’ interactions and integrates this history directly. Captured users’ interaction history from a drag-and-drop action to overall exploration enables to recall the insight of analyzed result. However, the captured history provides ordered exploration path with pre-defined user interactions but not encoded or abstracted interaction pattern. M. Pohl et al. [3] analyzed interaction sequence. They categorized user activities: select, explore, reconfigure, encode, abstract/elaborate, filter and connect. By the analysis of log files and thinking aloud method interaction sequence logs are extracted with transitional probabilities.

# Interaction Logging and Encoding

In this section, we describe how to capture users’ interaction logs and encode them to a specific format supporting recover and reuse. Interaction log data used here is manually gathered or created regarding limited time duration of this project.

## Interaction logs

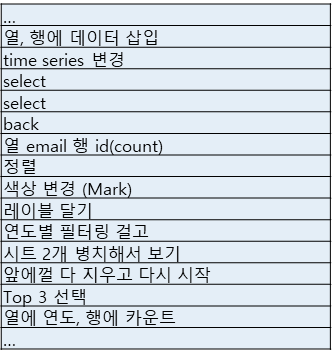
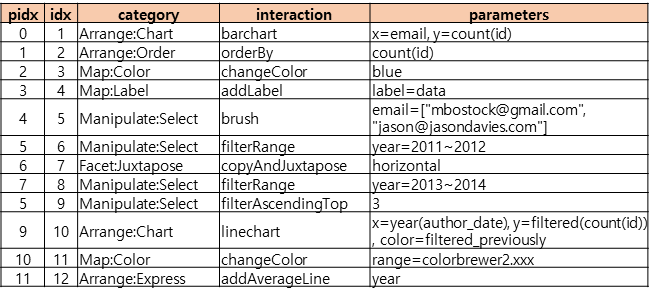
Analyzing software engineering data descripted in introduction, three participants (two senior developers, one junior developer) used commercial visual analytic tools (Spotfire, Tableau). In order to capture their interaction logs during analysis, the participants were encouraged to think aloud and write down their interactions on notes manually. The thinking aloud method was mainly intended to provide separation of each interaction and memorize serial interactions. An example of interaction logs written on note is figure 1(a).

## Encoding interaction logs: IR

Manually written interaction logs are necessary to be encoded as specific format which our visualization system is able to recognize. Instead of converting logs to the format directly, we first convert the logs to intermediate representation(IR) as modern compiler technique does [4]. The advantages of using IR are as follows.

* Common abstractions for diversity of the way each participant writing interaction logs
* Extensibility for coping with different visual analytic tools.
* Applying different encoding scheme for recover & reuse according to domain or purpose (in this project, we use only one scheme)

The structure of IR is depicted in figure 1(b). Basically, interaction category refers to Tamara’s visualization interaction idiom [5]. IR is converted to final format for visualization according to encoding scheme. Details of the format is described next section.

1. (b)
2. (a) An example of interaction logs written manually.   
   (b) An example of intermediate representation

After converting logs to IRs, we encoded it to a structured format for visualization. Whole reasoning process can be lengthy and complex, hence it is necessary to restructure sequential IRs as two level hierarchy: anchor and details. We first mark an anchor on the IR when meaningful change occurred as follows:

* Creation or alternation of kind of chart.
* Facet Alternation
* Data or range of data.

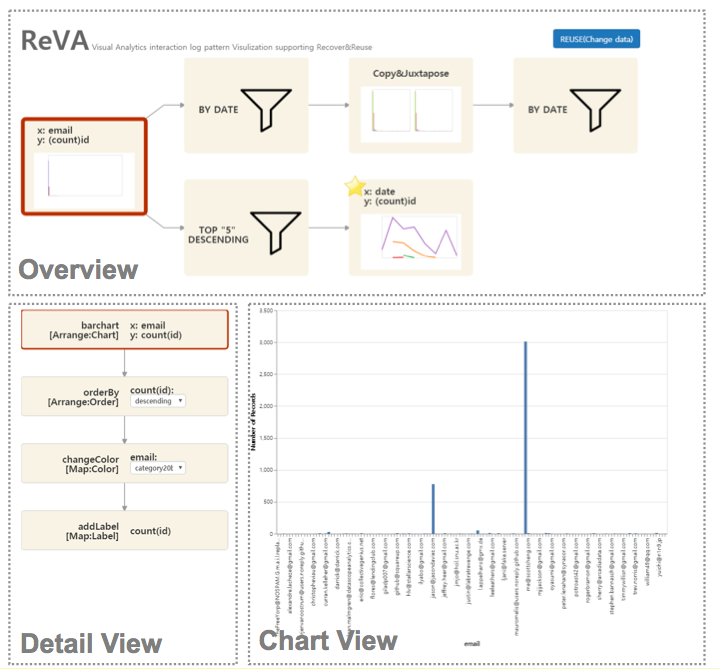
(For example, when an IR indicates that a bar chart is changed to a line chart or specific range of data is filtered, it goes to an anchor.) Then IRs, which are not anchors, are aggregated to previous anchor sequentially as figure 2.

1. Intermediate Representation Abstraction

# ReVA: interface supporting recover and reuse

We present ReVA, an interactive visualization interface for recovering and reusing interactions (Figure 3). Based on the idea with overview-first and focus-plus-context, we configure the layout as an overview, detail view, and chart view. In the overview, it shows people's reasoning process in simplified path form consisting of anchor nodes. We have set the standard to create a new anchor node whenever the chart changes shape or data changes. The detail view shows the detailed interactions contained within each anchor node. In the detail view, the user can customize the parameters included in each interaction. The chart view is linked with the detail node and provides a visualization corresponding to the detail interaction. ReVA also provides the ability to re-apply a user’s reasoning process to a new data or domain.

In the figure3, we set up a scenario that uses d3 commit data from Github. In this scenario, let us imagine a user who vaguely wants to get insight using github d3 commit data. REVA allows the user to find their own insights by looking at other users' reasoning process.



1. A ReVA showing two parallel paths that represent the reasoning processes of people.

# Pilot Study & Future work

We had three information visualization experts try the ReVA and get a brief review of its capabilities. Key comments from experts include: “In the chart view, you need to add a chart transition function whenever the chart changes according to interaction.” and “When drawing an anchor node in the overview, it may be more useful to vary the way the graph is represented (e.g., resize the nodes) depending on the frequency of the log pattern.”

Our further research is mainly focused on interaction log analysis. Tracking interaction log and gathering them are first-headed works for the research. Extracting interaction log patterns which are frequently used and encoding logs as context-aware type are our main goal. Refining more universal intermediate representation and abstraction of anchor graph are also to be considered.

# Conclusion

Our main contributions are as follows: 1) Define how to descript interaction logging pattern, 2) ReVA: Visual Interface that allows supporting recover and reuse.

References

1. C. North et al., “Analytic Provenance: Process + Interaction + Insight,” 29th Annu. CHI Conf. Hum. Factors Comput. Syst. CHI 2011, pp. 33–36, 2011.
2. Dunne, C., Henry Riche, N., Lee, B., Metoyer, R., & Robertson, G. (2012). GraphTrail: analyzing large multivariate, heterogeneous networks while supporting exploration history. CHI, Human Factors in Computing Systems, 1663-1663-1672–1672.
3. Pohl, M., Wallner, G., & Kriglstein, S. (2016). Using lag-sequential analysis for understanding interaction sequences in visualizations. International Journal of Human Computer Studies, 96, 54–66.
4. LLVM (https://llvm.org/)
5. Visualization Analysis and Design by Tamara Muzner