STEIN: speeding up evaluation activities with a Seamless Testing Environment INtegrator

M. Angelini[1] and G. Blasilli[1] and S. Lenti[1] and G. Santucci[1]

[1]Sapienza University of Rome, Rome, Italy

Abstract

The evaluation of an information visualization system is a complex activity, involving the understanding of both the visualization itself and the process that it is meant to support. Moreover, if the evaluation activity includes a task based user study, it requires a considerable effort, involving both conceptual (e.g., the definition of user tasks) and technical (e.g., logging of the relevant user actions while using the system) aspects. The solution presented in this paper, STEIN (Seamless Testing Environment INtegrator), allows integrating the system under evaluation with the questions that have been designed for the user study, tracing the user's activities and automatically collecting the user's answers using the events that are generated while interacting with the system. This results in a substantial reduction of the effort associated with technical activities, thus allowing the evaluation designer to focus mainly on the conceptual aspects. A prototype of the system is available for download at awareserver.dis.uniroma1.it:8080/stein.

CCS Concepts

•***Human-centered computing*** → ***User studies; Visualization systems and tools; Visualization toolkits; Visualization design and evaluation methods;*** *Laboratory experiments; Visual analytics; Information visualization;*

1. Introduction

The evaluation of information visualization systems is a complex activity since it not only involves assessing the visualizations themselves, but also the complex processes supported by the systems. Moreover, and this is the focus of the paper, if the evaluation includes a controlled experiment in which the user interacts with the system by performing tasks (see, e.g., [LBI*12]) and logs capturing user's actions (i.e., traces) (see, e.g., [IH01] or [HR00]), the situation is even worse. Indeed, it is required to collect the user's answers and the related low level details (e.g., response time, traces, etc.), further complicating the overall evaluation process. This paper copes with this problem by presenting STEIN (Seamless Testing Environment INtegrator) that allows integrating the system under evaluation (target system, in what follows) with the questions that have been designed for the user, tracing the user's activities and automatically collecting the user's answers using the events that are generated while interacting with the target system. After having embedded the target system by specifying its URI, STEIN allows the evaluation designer to easily extract relevant data types and events from it and to relate them to the questions of the evaluation and to the collected traces, thus supporting the main technical aspects of a system evaluation. The same environment allows executing the evaluation questions, automatically collecting user traces and answers. The main characteristics of STEIN are:

- full integration of the target system within the testing environment;
- automatic extraction of data types and events by just *interacting* with the target system;
- automatic traces collection of the events that have been selected by the evaluation designer;
- definition of the answers of the questionnaire in terms of target system data types;
- automatic answers collection through the user interactions with the target system;
- centralized access to remotely execute the evaluation.

It is worth noting that the proposed solution is mainly a technique for automating the execution of users' tasks on the target system collecting traces and answers, and we are not proposing it as *an evaluation methodology*; nor we claim that users' tasks and traces are the unique or the best way for performing an evaluation activity. Moreover, we do not assume that user's answers are always collectible by user's interactions with the system and STEIN allows also for getting answers directly from the user (e.g, a number, some text, etc.). Indeed, even if this is not the reason why we have developed it, STEIN can easily accommodate any kind of traditional textual questionnaire, using free text, or Likert and ratio scales (see, e.g., NASA TLX [HS88]), by just defining the questions, the associated scales, and the right answer intervals (or full text boxes for textual questions).

DOI: 10.2312/eurovisshort.20181083

This paper presents the rationale, the technical solutions, and the implementation of the STEIN system and it is structured as follows: Section 2 discusses related proposals, Section 3 describes the proposed system, Section 4 concludes the paper and highlights future work.

2. Related Work

The work in [LBI*12] discusses evaluation scenarios both for understanding data analysis processes and visualizations, and reviews the methods to assess them (e.g., controlled experiment, log analysis, etc.). Several works have faced with this topic, dealing with both general and specific aspects. The work in [Pla04] states how usability studies and controlled experiments are state-of-the-art evaluation methods but it points out the need to consider also other evaluation approaches, while the work in [SP06] asserts that the efficacy of a visualization system can be evaluated based on the usage of the system itself and expert users' success in achieving their professional goals. The purpose of the work in [Car08] is to increase awareness of empirical research and to encourage thoughtful application of a greater variety of evaluative research methodologies in information visualization.

Regarding the collection of user interactions, the work in [HR00] presents a survey on computer-aided techniques used by HCI researchers to extract usability-related information from user interface events. Vuillemot et al. in [VBT*16] aim to raise awareness of the potential of logging to improve visualization tools and their evaluation, as well as paving the way for a long term research agenda on the use of logs in information visualization, reporting a lack of methodology to support this process and to use the results consistently. The work in [PP02] presents a tool able to perform intelligent analysis of Web browser logs using the information contained in the user-defined task model of the application to evaluate the usability of generic web sites. To the best of the authors' knowledge the only two solutions that support a similar objective with respect to STEIN are Interaction Trace Manager and VisSurvey.js. Interaction Trace Manager [FB15] aims at supporting the collection of user interactions; it presents two main differences with respect to STEIN: a) it supports only the collection of traces while STEIN covers all the main aspects of the evaluation design and execution process, and b) its usage requires several low level activities (e.g., software installation, library import, coding inside the evaluated system, and running a SQL server to store the data) while STEIN requires only the URI of the system to execute the whole process. VisSurvey.js [JR17] is a tool that allows to create a user study of web-based systems rendering snapshots of the system into the evaluation environment and allowing to control the flow of the evaluation based on the answers given by the user; however, differently from STEIN it does not allow to interact with the system during the evaluation process.

3. Solution

The main goal of STEIN is to facilitate the evaluation process of an information visualization system. The design of the evaluation and the analysis of the collected results is out of the scope of this paper that, indeed, aims at providing a tool that supports this process leaving the evaluation designer free to structure the evaluation according to her needs. Generally, a user study evaluation comprehends a questionnaire on which users report the answers to the tasks they performed on the system, and the non trivial collection of traces. Additionally, in many cases the target system and the environment to answer the questionnaire are disjointed, pushing the user to switch context between them. In our opinion, this switch can lead to disruption in the work-flow, errors due to distractions and impacts the traces (e.g., answering time). The proposed solution aims at providing a seamless integration between the target system and the evaluation environment. STEIN supports the designer in the whole process, from the evaluation design to the collection of user traces, following the work-flow reported in Figure 1:

1. Target system embedding: to import the target system into STEIN by providing its URI;
2. Evaluation design: to design the evaluation process, further specified in:
   - System data types collection: to select (high-level) entities that the target system utilizes;
   - System events collection: to select the events that will be traced during the evaluation;
   - Questionnaire design: to manage the questionnaire structure and content;
   - Evaluation testing: to test the effectiveness of the evaluation environment during its creation;
3. Evaluation deploy: to distribute the evaluation and collect results, further specified in:
   - Evaluation running: to perform the evaluation process;
   - Tracing: to automatically collect users' answers, related events, and interactions with the target system.

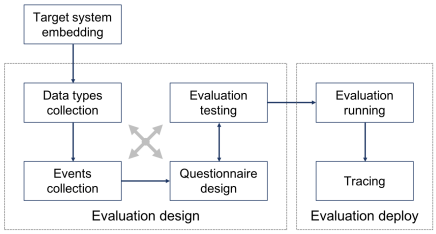

Figure 1: *STEIN main phases: after the embedding of the target system, the designer extracts the needed information from it using STEIN and arranges the questionnaire. While the users are conducting the evaluation process, their answers and interactions are recorded by STEIN.*

For each of these phases, STEIN provides a view, where most of them (both for the design and the deploy phases) are divided into a main panel (on the left) that contains the target system and a working panel (on the right) that manages the design choices in the design phase and the questionnaire in the running phase (see Figure 2).

3.1. System embedding

The first step is the integration of the target system into STEIN through its URI. The target system is encapsulated into an *iframe*,

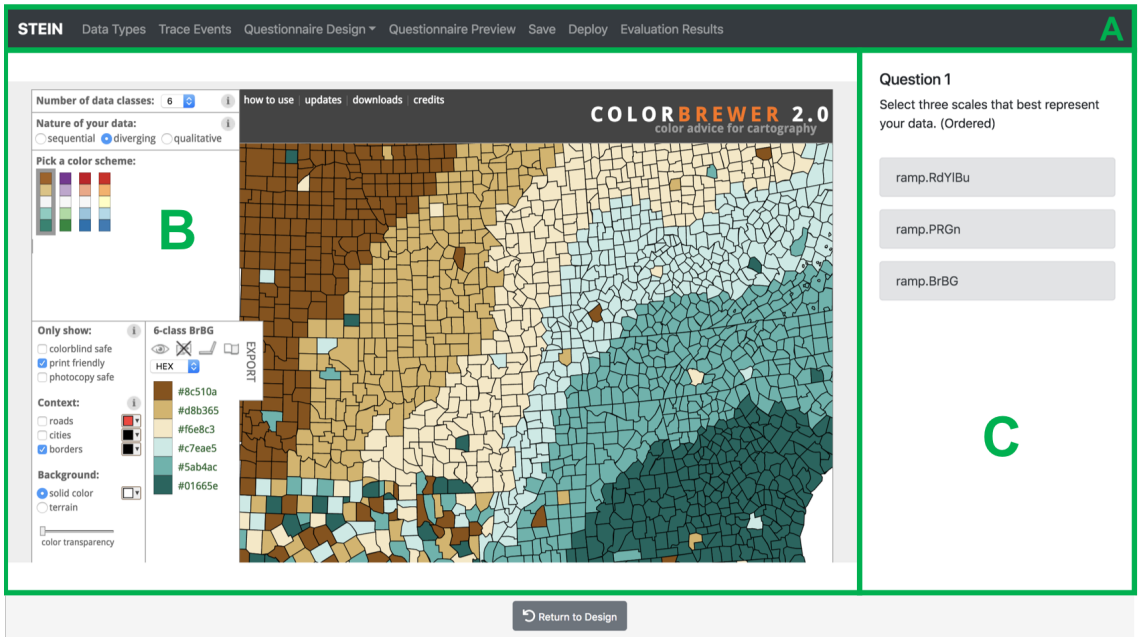

Figure 2: *STEIN question testing. The target system (ColorBrewer in this example) is embedded in the main panel on the left (B) and the working panel on the right (C) shows the question. The answer is automatically collected by selecting the scales in the target system. The menu bar on the top (A) allows to navigate between the different phases of the evaluation design.*

making it visible and interactive during the evaluation design in order to automatically collect target system properties. This allows conducting evaluations of not proprietary systems: for instance, STEIN has been tested against the popular tool ColorBrewer [HB03] that will constitute the use case for the rest of this paper.

3.2. System data types collection

Once the target system has been encapsulated into STEIN, the next step is the collection of its data types. A data type is a high level domain-dependent entity defined into the target system; as an example, in ColorBrewer a color scale is a data type. A data type can be used to fill an answer in the questionnaire and can be traced. STEIN collects all the data types defined in the target system during a pre-processing phase in which the user is asked to interact with the target system, and lists them in the right panel of the environment (see Figure 3). For expert users it is also possible to define additional complex data types and/or edit existing ones, all within the environment. Related to the use case, Figure 3 shows the collection of the data type "ramp" (a scale) of ColorBrewer. The information regarding the data type suggest that it is possible to consider the ramp for the following phases. Through the use of STEIN we identified 4 different data types for ColorBrewer (number of classes, scheme type, color system, and ramp).

3.3. System events collection

The next step is the collection of the events, which the designer wants to trace. Similar to data types collection, STEIN lists the collected events and their characteristics in the right panel. A generic event can have (e.g., click on a scale) or not (e.g., click on "colorblind safe" check-box) an associated data type. STEIN is able to collect both event types, suggesting, if appropriate, a link to previously collected data types. When an event is notified to the designer, s/he decides whether to add it to the list, optionally customizing the linked data type and the function to catch the event suggested by STEIN. All the events with an associated data type can be used within the seamless answering mechanism, automatically filling the user's answer with the associated data type. Regarding the complexity of events, STEIN is able to collect both simple DOM events (e.g., mouse-click, mouse-move) and complex

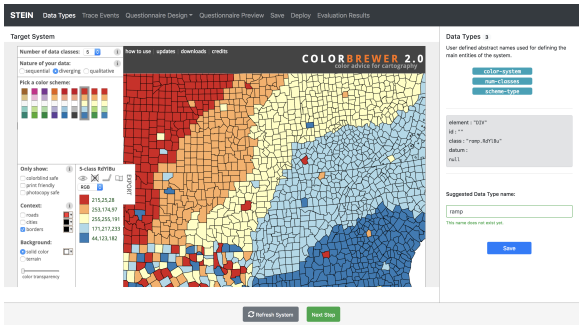

Figure 3: *System data types collection. Identification of the data type "ramp" suggested by STEIN and obtained by interacting with a color scale (a ramp) in the target system.*

ones (e.g., zoom, brush). The first ones are captured by overriding, in the target system, the *addEventListener* DOM method [Kac16]: first sending the event to STEIN and then calling the already defined listener function. Complex events, instead, are captured by watching changes on status event variables of DOM manipulation libraries (e.g., *d3.event* of the d3.js library [BOH11]). An API is provided to developers in order to communicate with STEIN and to better integrate the target system.

In the ColorBrewer use case, STEIN is able to capture 24 different events: 20 without an associated data type (e.g., change on blind-check, click on downloads, etc.) and 4 with an associated data type (e.g., click-on-ramp). Figure 4 shows the details of the collection of the scale selection event ("click-on-ramp"), triggered by clicking on one of the scales and linked to the ramp data type. At the end of this phase, the designer has collected (a subset of) the target system interactions and data types, having available the relevant data on which building the questionnaire.

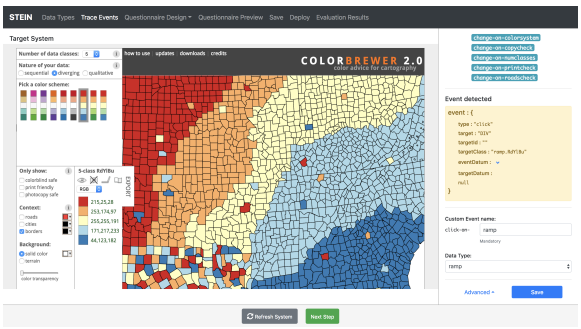

Figure 4: *The image shows the collection of the "scale selection event" linked to previously created "ramp" data type.*

3.4. Questionnaire design & evaluation testing

The STEIN system allows to define a questionnaire organized into three sections: initial questions, system questions, and final questions. Initial questions are proposed at the beginning of the evaluation to ask information not necessarily related to the target system (e.g., gender, age, etc.). The target system is not visible while showing these questions. System questions are strictly related to the target system and they are shown on the working panel; as a result, the user can interact with the system while answering the questions. Final questions are proposed at the end of the evaluation and they are, as the initial questions, not strictly related to the target system. For each question it is possible to select the desired response type (e.g., free text, multiple choices). System questions allow additional response types based on the data types and events collected in the previous phases: every time a collected event with the same data type of the question is dispatched, STEIN extracts the linked data type (e.g., a scale instance) and automatically adds it to the answer.

STEIN allows interactively testing each question at design time, in order to verify its efficacy, and if the response type is appropriate. At any time the designer can switch between the testing questions environment and the questionnaire design environment. Eventually, it is possible to test the whole questionnaire.

3.5. Evaluation running & tracing

At the end of the Evaluation design, the system generates a configuration file, in json format, used for deploying the evaluation. This file can be edited with STEIN at any time (e.g., to add questions, trace new events, define new data types). STEIN manages the persistence of the evaluation process, allowing the user to stop and resume the questionnaire at any time. At the end of an evaluation run, the system generates an output file (that can be exported to be processed for analysis in a different environment) containing the given answers and the traced events for each question. In addition to the collected events, STEIN traces a) the mouse movements and the list of all the elements that are hovered by the mouse during its movement and b) the response times, distinguishing between reading question time, and answering question time. The former is modeled as the time elapsed from the moment in which the question is shown until the generation of the first user's event on the target system. The latter is the remaining time until the user confirms the answer and moves on to the next question.

4. Discussion & Future Directions

During the development of an evaluation questionnaire we coped with the problem of creating an environment to design and run the process, generating the main requirements of the STEIN system. We highlight as main advantage of this approach the possibility to speed up the evaluation process design by not asking the designer to re-implement each time the technical infrastructure from scratch, allowing more time on the questionnaire design and improving the reuse of best practices from past evaluation activities. This is reinforced by the capability of concentrating all the evaluation activities in a single environment. STEIN targets different kinds of users, such as domain experts that do not have particular programming skills by providing an environment that captures the default behavior of a target system, not asking to cope with its code; nonetheless STEIN helps also the skilled programmers who may want to inject very specific behaviors to the target system (e.g., for defining additional events) by allowing their definition inside STEIN. We have used STEIN for our internal evaluation activities, i.e., evaluating a complex visual analytics system and comparing different implementations of a basic interaction activity, confirming that the seamless way of selecting the answers within the target system facilitates the user in answering the questionnaire and allows for obtaining more precise time tracking (in some cases the answer was compound by a list of items). Regarding possible use scenarios, in addition to the evaluation of proprietary systems, STEIN allows to conduct comparative analysis among several systems by embedding them in the same common evaluation environment, without inspecting their source code, like image visual encodings recovery in [PH17] [PMH18] [BDM*17]. About limitations, STEIN works only with Web based systems and pure desktop programs cannot be evaluated using it. We plan to further improve STEIN by implementing automatic events and data types extraction functionality that can assist the designer in the data types and events collection phases. We further plan to add a specific environment for trace analysis, allowing to use its results as a feedback during the evaluation design phase.

References

[BDM*17] BATTLE L., DUAN P., MIRANDA Z., MUKUSHEVA D., CHANG R., STONEBRAKER M.: Beagle: Automated extraction and interpretation of visualizations from the web. *arXiv preprint arXiv:1711.05962* (2017). 4

[BOH11] BOSTOCK M., OGIEVETSKY V., HEER J.: D3: Data-driven documents. *IEEE Trans. Visualization & Comp. Graphics (Proc. InfoVis)* (2011). URL: http://vis.stanford.edu/papers/d3. 4

[Car08] CARPENDALE S.: Evaluating Information Visualizations. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics) 4950 LNCS* (2008), 19–45. doi:10.1007/978-3-540-70956-5_2. 2

[FB15] FEKETE J.-D., BOY J.: Interaction trace manager, Apr 2015. URL: https://github.com/INRIA/intertrace. 2

[HB03] HARROWER M., BREWER C. A.: Colorbrewer. org: an online tool for selecting colour schemes for maps. *The Cartographic Journal 40*, 1 (2003), 27–37. 3

[HR00] HILBERT D. M., REDMILES D. F.: Extracting Usability Information from User Interface Events. *ACM Computing Surveys 32*, 4 (2000), 384–421. doi:10.1145/371578.371593. 1, 2

[HS88] HART S. G., STAVELAND L. E.: Development of nasa-tlx (task load index): Results of empirical and theoretical research. In *Advances in psychology*, vol. 52. Elsevier, 1988, pp. 139–183. 1

[IH01] IVORY M. Y., HEARST M. A.: The state of the art in automating usability evaluation of user interfaces. *ACM Comput. Surv. 33*, 4 (Dec. 2001), 470–516. URL: http://doi.acm.org/10.1145/503112.503114, doi:10.1145/503112.503114. 1

[JR17] JACKSON J., ROBERTS J.: Vissurvey.js - a web based javascript application for visualisation evaluation user studies. In *Poster presented in 2017 VIS IEEE Conference* (2017). URL: https://github.com/jamesjacko/visSurvey. 2

[Kac16] Ui events, w3c working draft, Aug 2016. URL: https://www.w3.org/TR/DOM-Level-3-Events/. 4

[LBI*12] LAM H., BERTINI E., ISENBERG P., PLAISANT C., CARPENDALE S.: Empirical Studies in Information Visualization: Seven Scenarios. *IEEE Transactions on Visualization and Computer Graphics 18*, 9 (2012), 1520–1536. doi:10.1109/TVCG.2011.279. 1, 2

[PH17] POCO J., HEER J.: Reverse-engineering visualizations: Recovering visual encodings from chart images. In *Computer Graphics Forum* (2017), vol. 36, Wiley Online Library, pp. 353–363. 4

[Pla04] PLAISANT C.: The Challenge of Information Visualization Evaluation. *Proceedings of the working conference on Advanced visual interfaces - AVI '04* (2004), 109. URL: http://portal.acm.org/citation.cfm?doid=989863.989880, doi:10.1145/989863.989880. 2

[PMH18] POCO J., MAYHUA A., HEER J.: Extracting and retargeting color mappings from bitmap images of visualizations. *IEEE transactions on visualization and computer graphics 24*, 1 (2018), 637–646. 4

[PP02] PAGANELLI L., PATERNÒ F.: Intelligent Analysis of User Interactions with Web Applications. *Proceedings of the 7th international conference on Intelligent user interfaces - IUI '02* (2002), 111–118. URL: http://dl.acm.org/citation.cfm?id=502735{%}5Cnhttp://dl.acm.org/citation.cfm?id=502716.502735, doi:10.1145/502716.502735. 2

[SP06] SHNEIDERMAN B., PLAISANT C.: Strategies for Evaluating Information Visualization Tools: Multi-dimensional In-depth Long-term Case Studies. *Proceedings of the 2006 AVI workshop on BEyond time and errors: novel evaluation methods for information visualization - BELIV '06* (2006), 1–7. URL: http://portal.acm.org/citation.cfm?id=1168149.1168158, doi:10.1145/1168149.1168158. 2

[VBT*16] VUILLEMOT R., BOY J., TABARD A., PERIN C., FEKETE J.-D.: Challenges in Logging Interactive Visualizations and Visualizing Interaction Logs. In *Workshop on Logging Interactive Visualizations and Visualizing Interaction Logs (LIVVIL '16)* (Baltimore, United States, 2016). 2