Command History as a Full-fledged Interactive Object

Objectives and Positioning

Histories of command and their functionalities have undergone little evolution since their initial design, the ubiquitous "undo-redo", decades ago. In most applications today, the commands that contributed to a document are merely used as labels for intermediate document versions; all the information used in their computation is forgotten as soon as the command is completed, and so are "undone" commands after they are replaced by other commands. This PhD project stems from the observation that this information is under-used, and forgotten too systematically; it aims at greatly increasing the navigating and editing capability of such systems. **The goal is to treat the user's actions as full-fledged interactive objects**, thus allowing users to edit and manipulate not only the most recent version of a document, but also its whole creation process. The thesis will propose new ways to navigate and exploit these augmented histories for content editing, error correction, and creative process sharing. The outcome of this project will be to augment the flexibility and operational vocabulary of interactive systems by including complete and interactive command histories at the lowest level of their design and use.

Academic and Industrial Context

Command histories (undo/redo) are often only composed of a subset of a document's versions, labelled by the name of the preceding command. This allows basic navigation and error correction in the document's chronology, but these functionalities remain limited and can cause severe information loss [1]. Most of the information that was known and used at the time of the command is forgotten after execution, whereas it could be stored and linked for later access, sharing, reuse, and even *a posteriori* modification.

Numerous new and augmented features for command histories have been proposed in the Human-Computer Interaction (HCI) literature: parallel timelines [2], undoing actions that are not the most recent [3], undoing commands based on their location in the document [4], etc. However, these techniques were designed and evaluated independently, and are usually incompatible with each other—sometimes even with basic undo/redo! Consequently, they are almost never implemented in real applications.

We already proposed a unified model of interaction history in [1], which allows great theoretical flexibility in navigating, reusing and modifying past commands, as well as their parameters and the input involved in their computation. This model identifies and solves some of the theoretical flaws and functional contradictions in previous approaches. It encompasses the new functionalities that they proposed, and allows new editing functionalities to emerge. Part of the PhD project is to apply this model to real applications in order to evaluate its potential for interaction and as a design pattern.

The PhD Project

The PhD will start with a comprehensive literature review on command histories, that will cover at least three topics: (i) advanced features and functionalities of command histories, (ii) visualization and interaction techniques to understand, navigate, and manipulate these histories, and (iii) their software engineering. This literature review will consolidate or improve on the model proposed in [1], in order to lay the conceptual foundations for its implementation.

The core of the PhD project can follow three major axes:

(1) Design, implementation, and evaluation of a **data structure** to log the user's actions with enough detail that they can be replayed identically: the command, its parameters, input and context, the elements to which it is applied, and its consequences. Once this information is chained, the whole design process of a document can be replayed, and any aspect of its history can be accessed, reused, and even modified programmatically.

This axis will require significant software engineering and design: the data structure must be implemented at a low level, and flexibly enough that new visualization and manipulation techniques can be implemented on top of it. It should also be implemented in realistic applications to assess real implementation difficulties and trade-offs depending on the type of manipulated data (e.g. images, videos, text, etc.). Finally, it should allow graph manipulations such as timeline branching and various advanced requests on command history.

(2) Design and evaluation of **visualization and interaction techniques** to navigate, reuse and manipulate the command histories, as clearly and efficiently as possible for the user. These techniques can be defined at various levels of abstraction and granularity of the underlying model, to allow a progressive mastering of its features. Specific attention will be given to skill acquisition and to the needs of distinct user populations: some might be satisfied with augmented error recovery, others be interested in specific functionalities without wanting to manipulate the lowest-level elements of the model, and so on. It will therefore be necessary to design techniques to navigate and manipulate every aspect of the history of a document, but also to select higher-level operations on history to be presented as independent functionalities.

The proposed techniques will be evaluated on criteria such as immediate understandability, software and human performance, skill acquisition and transitioning to advanced functionalities, the emergence of original uses, etc. The difficulty will be to help users conceptualize novel concepts efficiently, and to make these concepts clear and efficiently controllable depending on the type of manipulated data.

(3) Definition of **software engineering methods** to apply and generalize the results obtained above to different types of applications. The goal is to propose new software engineering norms that integrate the principles defined and validated during the thesis, via e.g. dedicated libraries, new design patterns, or even defining new programming keywords to automate some of the model's functionalities at language level (e.g. declaring that a method must maintain a history of its calls, of their parameters, and of what they affected).

Candidates and Context of the PhD Thesis

The candidate needs strong skills in programming and software engineering, know the basics of HCI (participatory design, controlled experiments) and be creative.

Funding: 3 years expected (ANR JCJC "Causality")
Lab(s): Inria Lille – Nord Europe, LOKI group.
Advisors: Mathieu Nancel (Mathieu.Nancel@inria.fr) Stéphane Huot (Stéphane.Huot@inria.fr).

[1] Nancel, M. and Cockburn, A. CAUSALITY – A Conceptual Model of Interaction History. ACM CHI ’14.
[2] Terry, M. et al. Variation in element and action: supporting simultaneous development of alternative solutions. ACM CHI ’04.
[3] Prakash, A. et al. A framework for undoing actions in collaborative systems. ACM ToCHI (1994).
[4] Seifried, T., Rendl, C., Haller, M., and Scott, S. Regional undo/redo techniques for large interactive surfaces. ACM CHI ’12.