VizScribe: A Visual Analytics Approach to Understand Designer Behavior

Senthil Chandrasegaran[1], Sriram Karthik Badam[2], Lorraine Kisselburgh[3,4], Kylie Peppler[7], Niklas Elmqvist[1,2] and Karthik Ramani[5,6]

[1]*College of Information Studies*
[2]*Department of Computer Science*
University of Maryland, College Park, MD, USA
[3]*The Center for Education and Research in Information Assurance and Security*
[4]*The Burton Morgan Center for Entrepreneurship*
[5]*School of Mechanical Engineering*
[6]*School of Electrical and Computer Engineering (by courtesy)*
Purdue University, West Lafayette, IN, USA
[7]*School of Education*
Indiana University, Bloomington, IN, USA

Abstract

Design protocol analysis is a technique to understand designers' cognitive processes by analyzing sequences of observations on their behavior. These observations typically use audio, video, and transcript data in order to gain insights into the designer's behavior and the design process. The recent availability of sophisticated sensing technology has made such data highly multimodal, requiring more flexible protocol analysis tools. To address this need, we present VizScribe, a visual analytics framework that employs multiple coordinated multiple views that enable the viewing of such data from different perspectives. VizScribe allows designers to create, customize, and extend interactive visualizations for design protocol data such as video, transcripts, sketches, sensor data, and user logs. User studies where design researchers used VizScribe for protocol analysis indicated that the linked views and interactive navigation offered by VizScribe afforded the researchers multiple, useful ways to approach and interpret such multimodal data.

Keywords: Protocol analysis, design research, design behavior, human-computer interaction, information visualization, visual analytics

1. Introduction

Design as an activity can be interpreted in different ways: as a social practice that requires interaction between designers (Oak, 2011), as a cognitive process in the form of a dialogue between the designer and their sketch (Goldschmidt, 1991), or as an elicitation of tacit knowledge by the designer (Henderson, 1998). In all, action, interaction, and communication are essential components of the designer's process. To understand the designer's thinking and the design process, it is necessary to study how designers interact with each other and their environment.

Such studies, often falling under the broader category of *protocol studies* involve recording the design session on video and audio, followed by transcribing, segmenting, and coding such data. Dinar et al. (2015) discuss such analyses in their detailed review of empirical studies of design, including traditional protocol analyses that involve coding designer utterances and actions. Typically verbal data forms the basis of such analyses, such as think-aloud protocol analysis where a designer verbalizes her activities, giving insight into her thought process (Ericsson, 2006), or latent semantic analyses of a design team discussion to assess coherence between team members (Dong, 2005).

Analysis of verbal data requires manual transcriptions of participant verbalizations by transcribers—often lacking domain knowledge—provided with a relevant glossary of terms. These transcriptions and segmentations then need to be verified by employing multiple coders. The design researcher may not be the one coding the data, although they may need to explore the data to identify questions that need answering. In addition to audio and video data, protocol studies often also involve analysis of sketches, notes, and other "marks-on-paper" (Ullman et al., 1990, p.269) that are integral to designer behavior. To designers, sketches function as external memory, or as a medium to think. To analysts, sketches are externalizations of the designer's thoughts, providing insights into their mind. Sketches are also logs of their thought sequences.

Qualitative analyses of protocol studies—both in-situ and in-vivo—may require making sense of multimodal data. This data can include, but is not restricted to, audio/video recordings, artifacts created by subjects, or electronic records of user activity, typically through the use of *Computer-Aided Qualitative Data Analysis Software* (CAQDAS). However, while there are many options in CAQDAS software for analyzing and coding video, transcripts, and document data, there are few that provide support for broader forms of data, including movement, activity and position data from smartphones or sociometric sensors, or even psychophysiological measurement devices such as electroencephalography. Dinar et al. (2015), in their review of 25 years of research in design theory and methodology, suggest the use of automated data collection and analyses to process the increasing volume and heterogeneity of data collected in such studies.

To address these challenges, we look to the field of visual analytics for inspiration. Visual analytics is "the science of analytical reasoning facilitated by interactive visual interfaces" (Thomas & Cook, 2005, p. 4). Specifically, it focuses on interactive visual representations that exploit the human ability to spot patterns and anomalies, while allowing the computer to process large datasets. Information visualization (Infovis), defined as a "graphical representation of data or concepts" (Ware, 2012, p.2), naturally forms an essential part of visual analytics. Keeping these needs and possibilities in mind, we present VizScribe, a web-based framework that supports the representation of multimodal data, including traditional as well as newer forms discussed earlier. The framework allows analysts to generate appropriate visual representations of temporal datasets, and link them to the existing video and transcript displays. These visualizations include line or area charts, timeline plots, color maps, or any other visualization that the user can construct using the underlying JavaScript library. VizScribe supports interactive, coordinated visualizations that can be customized. The coordination between visualizations is performed through brushing and linking (Li & North, 2003), an infovis technique that, when the user selects a part of the data shown on visualization, updates the other coordinated views to reflect the selection. The technique allows analysts to easily and intuitively explore the data and interactively code it. In the context of protocol analysis, this supports contextual exploration of data, allowing the analyst to attend not only to singular data sources, but also intersectional and contextual factors. This in turn helps analysts develop a "thick description" (Geertz, 1973) of the designer's behavior: a description that explains both the mechanics and the context of the behavior.

The contributions of this work are twofold: (1) a web framework for protocol analysis inspired by visual analytics techniques, and (2) results from formative and summative studies of protocol analysts using the framework to analyze a team brainstorming session. The framework itself enables the analyst to:

- Explore and analyze time-stamped verbal protocol data by creating coordinated, interactive timeline visualizations, word-cloud views, and transcript views that are linked to the corresponding video data,

- Visualize other protocol data such as server logs and biometric/sociometric sensor data in the form of interactive timeline visualizations, linked to the visualizations of the verbal protocol data above,

- Select and code the verbal protocol data, and finally
- Customize or add new timeline visualizations that are pertinent to their study.

To test the usefulness and to refine the usability of the VizScribe framework, we performed two studies: (1) a formative study involving both prescribed and open coding tasks, and on further refining the framework, (2) a summative study in which participants analyzed a 60-minute design session through video, transcript, sketch, speech, and activity data. Our studies demonstrated that timeline and text views of the transcript, and the timeline view of sketching activity were the most widely-used visualizations for most tasks, while word cloud visualizations were used as a filter to identify salient parts of the transcript. We observed that participants both with and without training in CAQDAS tools were comfortable using VizScribe to sift through the data. VizScribe allowed analysts to orient themselves with a medium that they found most comfortable—sketches, transcript, or the video of the design session shown—with little difference in the task outcome or the user feedback.

In the following sections, we review related work in protocol studies, motivate the need for visualization techniques to represent both temporal and text data, and detail the design and implementation of the VizScribe framework. We then describe the user studies, and participant feedback, finally highlighting challenges and future work.

2. Background

In this section we discuss existing tools for protocol studies. We then motivate the need for two kinds of visualization techniques for such studies: event-based representations for temporal analysis, and text visualizations for inferring patterns in the structure and semantics of the transcribed text.

2.1. Existing tools for protocol studies

Early work on computational support for protocol studies included artificial intelligence-based systems such as PAS-II (Waterman & Newell, 1973), which incorporated linguistic processing on task verbalizations to generate graph representations of human knowledge, and KRITON (Diederich et al., 1987), a knowledge-extraction system that infers knowledge elements and then forms relations using propositional calculus. Other approaches to support such studies used generic software such as AQUAD (Huber & Garca, 1991) and SHAPA (Sanderson et al., 1989) for collecting and organizing data, metadata, and annotations.

Commercial tools for qualitative analysis include multimedia processing tools such as ATLAS.ti[1] and NVivo[2]. These allow visualization and annotation of video and transcribed text, preserving associations between the two. The Computational Analysis Toolkit (Lu & Shulman, 2008) extends ATLAS.ti to a web-based framework while increasing coding flexibility and speed. StudioCode[3] is another popular tool for video and transcript analysis. Other tools, such as LINKOgrapher (Pourmohamadi & Gero, 2011) are developed for more downstream applications: analysis of codes using a predefined ontology in the context of conceptual design. However, these tools cannot provide coordinated views of multimodal data, nor can they adapt to diverse and evolving forms of data that protocol studies are beginning to entail. VizScribe is designed for integrating of multimodal data, allowing a synchronized view of all recorded events accompanying the main audio/video and transcript, thus providing context to the user.

Each of these tools has their specific advantages and disadvantages; for instance, there are issues such as forced workflows (imposing a specific style and sequence in coding), coding fetishism (using coding irrespective of whether or not it is appropriate), distancing the user from data, or, at the other end of the spectrum, difficulty in conceptual abstraction (Duff & Séror, 2005). Complete automation of the coding process is another challenge. For example, automated coding tools need better approximations of linguistic features inherent to coding schemes (Rosé et al., 2008). While it is unclear whether a visual analytics-based approach will mitigate some of these issues, a combination of visual representation and computational analysis should serve to augment the pattern-recognizing strengths of the human user and the data processing power of the computer.

2.2. Visualizing Event-Based Data

Analysis of data collected from protocol studies involves a temporal component, with a view to identifying and annotating co-occurring events. We restrict this section to related work in timeline visualizations of categorical data, which is more relevant to protocol studies than numerical data.

Lifelines (Plaisant et al., 1996) is a general representation of biographical data, visualizing discrete events and relationships, and allowing focus on specific parts on the timeline information. Wang et al. (2008) use timeline-based interactive visualizations to align events in patient medical records in order to identify

[1]`http://www.altasti.com/`

[2]`http://http://www.qsrinternational.com/`

[3]`http://www.studiocodegroup.com/`

co-occurrences of other related events. Challenges in interacting with such representations are illustrated by Monroe et al. (2013), who developed a visual query mechanism for these events.

CyberForensic Timelab (Olsson & Boldt, 2009) uses timeline displays for establishing history in cyberspace. It displays a timeline view of electronic records of personal, time-stamped events, to provide the investigator with a visual history of events. PortVis (McPherson et al., 2004) uses an overview and detail-on-demand approach to display activity on a large set of TCP ports in a network over time, identifying traffic anomalies that signify possible attacks on the network.

Stab et al. (2010) use timeline views for more general applications: they develop SemaTime, a temporal visualization tool that allows hierarchical categorization and filter of domain-specific data. It also incorporates semantic relationships between entities, similar to Continuum (André et al., 2007), which also provides histogram overview and timeline detail view of temporal events. Temporal relationships are represented as spans of bounding boxes, providing a visually pre-attentive visualization, with the level of detail controlled using a "dimension filter". Rubin et al. (2013) present a set of tools for Navigation and editing between speech and transcript for high-level editing of interviews to create "audio stories". These tools, though not meant for protocol studies, use methods that are both relevant and useful for future iterations of VizScribe.

More relevant to our work are the Digital Replay System (Brundell et al., 2008), an ethnographic tool that uses an ontology-based data representation for multimodal data analysis, and Chronoviz (Fouse et al., 2011), a timeline-based annotation and visualization of multimedia data that uses timestamps to display video, audio, and electronically-recorded annotations with related timeline data such as geospatial data and sensor logs. Like Chronoviz, VizScribe provides a coordinated view of multiple timeline visualizations, and a way for the user to modify and create custom timeline visualizations. However, VizScribe differs from Chronoviz in three main ways:

- It emphasizes verbal data by providing three different, mutually coordinated views of the transcript: a timeline view, a text view, and a word cloud view for temporal and overview exploration, and finally a fourth "text concordance" view of all the contexts in which a selected word is used in the data.

- It supports the creation and use of more complex timeline-based visualizations such as sketching behavior which is partly temporal (e.g. sketch creation times) and partly semantic (e.g. showing the progression of a sketch and its collaborative development over time).

- Finally, the framework runs on the web browser and uses the D3 visualization plugin, making it both cross-platform and customizable.

While Dedoose is also a web-based and cross-platform tool, the remaining advantages of our framework are maintained when compared to it, as well as to other CAQDAS tools. However, it is worthwhile to note that VizScribe is an implementation of a visual analytics *approach* to protocol analysis, and not an end-to-end CAQDAS product. As such, the focus of this work lies in the rationale behind this approach, its design, implementation, and evaluation.

2.3. Visualizing Text

Protocol analysis almost always includes transcribed text as a main data format. These are thus uniquely suited for a combination of text visualization linked to associated timeline visualizations discussed earlier. Previous tools for in-situ studies have used computational linguistics to draw inferences. However, there is no panacea for cross-domain linguistic analysis. Polysemy and sentence parsing issues make a completely automated text analysis tool a considerable challenge.

Developments in visualization techniques have opened up another dimension in text analysis: visualizing text data. Basic text visualizations include frequency-based word clouds such as Wordle[4], keyword in context representations (Manning & Schütze, 1999), and lexical dispersion plots (Hoey et al., 2001). More sophisticated visualizations involve a degree of aggregate representation, or representation of metadata. The Word Tree (Wattenberg & Viégas, 2008) is an example of the former, with its aggregation of concordant terms to form a 'tree' of words or phrases, scaled by occurrence. Parallel Tag Clouds (Collins et al., 2009b) is an example of the latter, with tag clouds in the form of parallel axes to represent relationships between multiple documents.

Representations for document content overview include Arc Diagrams (Wattenberg, 2002), which represent document structure visually, or in semantic form as visualized in DocuBurst (Collins et al., 2009a). Such semantic bird's-eye visualizations work well when combined with the more detailed keyword visualizations for an effective combination of overview and detail. VizScribe uses the word cloud and keyword in context representations, and a version of lexical dispersion plots as the transcript timeline view with interactive linking between these.

[4]`http://www.wordle.net/`

3. Design Rationale: Visualizing Multimodal Protocol Data

Newell (1966, p.1) defines a protocol as “a record of the time sequence of events” that also includes “continuous verbal behavior of the subject” in the context of problem solving in a think-aloud setting. These transcripts of verbal behaviors are studied, annotated, and coded by design researchers, in order to form and test hypotheses, and to answer research questions (Pourmohamadi & Gero, 2011). Such annotations and codes are created by first identifying design behaviors situated in a specific time and context, determining categories of such behavior, and assigning them to appropriate codes. The analyst’s challenge is to make sense of a multitude of such time series data or observations, by linking them to transcripts of designer discourses, video captures of designer actions, and artifacts such as sketches or prototypes created by the designer.

Our layout design and the interactions afforded by our framework are informed by these aspects of protocol studies. With the increasing prevalence in the use of software tools for design and the use of sensors such as accelerometers, gyroscopes, and bluetooth devices to track collaboration dynamics in teams (Kim et al., 2012), protocol data have become richer and more varied. Making sense of such multiple forms of data depends on two major factors: the way in which these datasets are displayed to the user (analyst) and the way in which the user is able to interact with these datasets. Both of these requirements are commonly encountered when designing infovis and visual analytics tools and will thus benefit from principles derived from these fields.

VizScribe is designed essentially as a cohesive collection of *coordinated multiple views* (Heer & Shneiderman, 2012, p.12)—linked views of related datasets that enable the viewing of the data from multiple perspectives. The rationale and requirements behind this framework are detailed in this section.

R1 *Spatial Coherence:* Visually representing time sequences of events, including verbal behavior, needs a spatially coherent alignment to make sense of these sequences. Position is a widely accepted encoding of quantities occurring on a scale (Carpendale, 2003), the scale in this case being time. All timeline data pertinent to a video-recorded protocol study session must thus be presented coherently with the video timeline.

R2 *Multiscale Representation:* The substantial data obtained from protocol studies can benefit from multiple scales of representation: of time, content, and detail. These allow the user to identify patterns and form meaningful connections between data. This “overview + detail” approach (Shneiderman,

1996) is a fundamental requirement of interactive visualizations, and is integral to the interpretive coding process as well (Creswell, 2012).

R3 *Context Awareness:* Connections between different representations of data need to be made evident to the user as needed. For data from protocol studies, this is especially important in order to provide context. This is also useful when the analyst is trying to understand if there are patterns of behavior evident in the coded sets of data.

R4 *Interactivity:* Making sense of multiple datasets requires fluid interactions with their representations. These interactions should allow the user to *orient* themselves to data visualizations, *focus* on specific elements of the visualizations, and then *code* the data. Elmqvist et al. (2011, p.336) emphasize the importance of "providing immediate visual feedback on interaction" for both major and minor interaction events. They also recommend allowing the user to interact directly with the visual elements.

R5 *Extensibility:* Given the ever-changing nature of data collected in protocol studies, such visualizations need to be extendable to accept newer forms of data, mapped to appropriate visual representations.

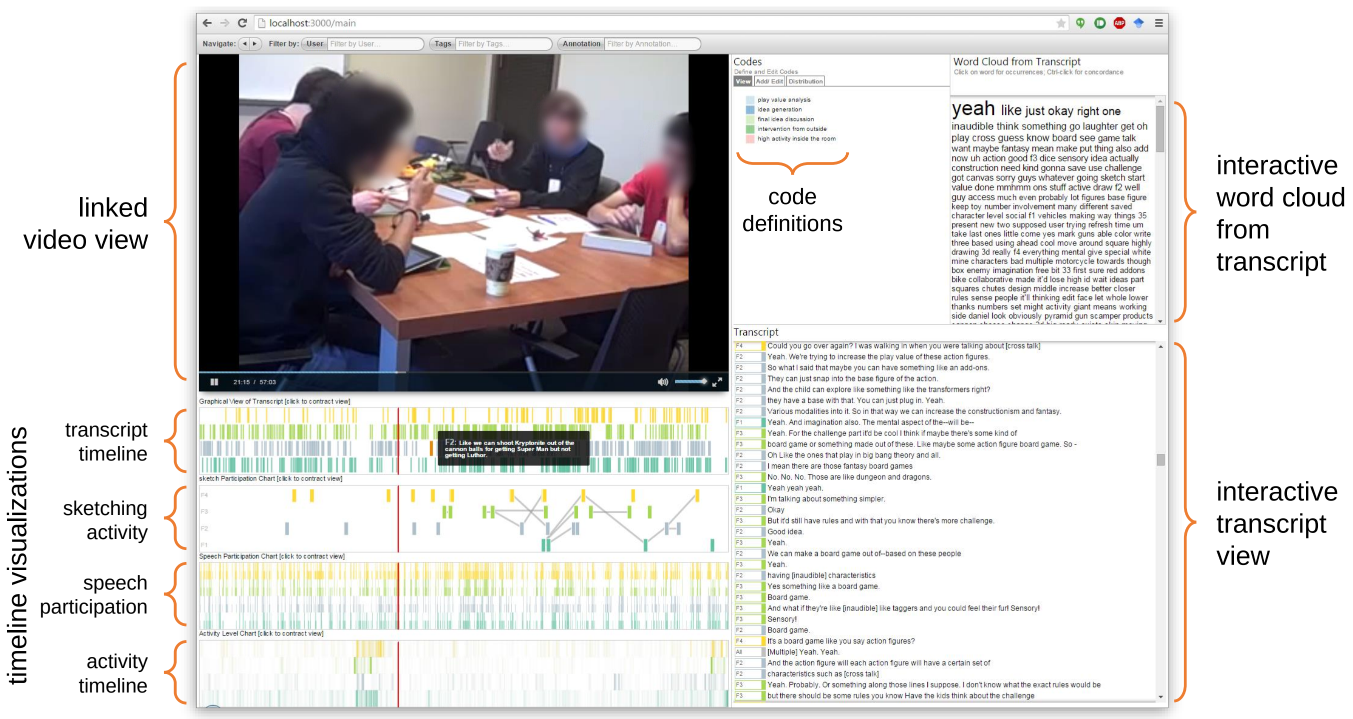
4. The VizScribe Framework

VizScribe's design addresses the previously-established requirements of spatial coherence, multiscale representations, context awareness, interactivity, and extensibility. The two main aspects of this design are the *layout*—determining how the data is represented to the user— and *interactions*—determining how the user explores the data. We detail both these aspects here.

4.1. Layout

VizScribe's layout is divided into two main sections, shown in Figure 1:

- A *temporal view*, where all time-series data is shown in congruence with the video of the design session being studied, and
- A *transcript view* where the transcript and other transcript-related data including word frequencies and codes are shown.


localhost:3000/main
Navigate:
Filter by:
User
Filter by User...
Tags
Filter by Tags...
Annotation
Filter by Annotation...
21:15 / 57:03
Graphical View of Transcript [click to contract view]
F2: Like we can shoot Kryptonite out of the cannon balls for getting Super Man but not getting Luthor.
sketch Participation Chart [click to contract view]
Speech Participation Chart [click to contract view]
Activity Level Chart [click to contract view]
Codes
Define and Edit Codes
View
Add/ Edit
Distribution
play value analysis
idea generation
final idea discussion
intervention from outside
high activity inside the room
Word Cloud from Transcript
Click on word for occurrences; Ctrl-click for concordance
yeah like just okay right one inaudible think something go laughter get oh play cross guess know board see game talk want maybe fantasy mean make put thing also add now uh action good f3 dice sensory idea actually construction need kind gonna save use challenge got canvas sorry guys whatever going sketch start value done mmhmm ons stuff active draw f2 well guy access much even probably lot figures base figure keep toy number involvement many different saved character level social f1 vehicles making way things 35 present new two supposed user trying refresh time um take last ones little come yes mark guns able color write three based using ahead cool move around square highly drawing 3d really f4 everything mental give special white mine characters bad multiple motorcycle towards though box enemy imagination free bit 33 first sure red addons bike collaborative made it'd lose high id wait ideas part squares chutes design middle increase better closer rules sense people it'll thinking edit face let whole lower thanks numbers set might activity giant means working side daniel look obviously pyramid gun scamper products
Transcript
linked video view
code definitions
interactive word cloud from transcript
interactive transcript view
timeline visualizations
transcript timeline
sketching activity
speech participation
activity timeline

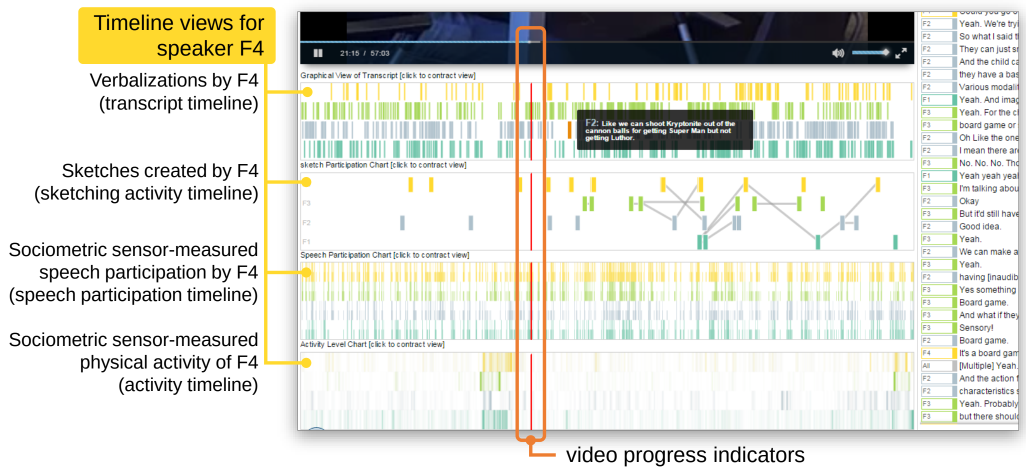
F4	Could you go over again? I was walking in when you were talking about [cross talk]
F2	Yeah. We're trying to increase the play value of these action figures.
F2	So what I said that maybe you can have something like an add-ons.
F2	They can just snap into the base figure of the action.
F2	And the child can explore like something like the transformers right?
F2	they have a base with that. You can just plug in. Yeah.
F2	Various modalities into it. So in that way we can increase the constructionism and fantasy.
F1	Yeah. And imagination also. The mental aspect of the--will be--
F3	Yeah. For the challenge part it'd be cool I think if maybe there's some kind of
F3	board game or something made out of these. Like maybe some action figure board game. So -
F2	Oh Like the ones that play in big bang theory and all.
F2	I mean there are those fantasy board games
F3	No. No. No. Those are like dungeon and dragons.
F1	Yeah yeah yeah.
F3	I'm talking about something simpler.
F2	Okay
F3	But it'd still have rules and with that you know there's more challenge.
F2	Good idea.
F3	Yeah.
F2	We can make a board game out of--based on these people
F3	Yeah.
F2	having [inaudible] characteristics
F3	Yes something like a board game.
F3	Board game.
F3	And what if they're like [inaudible] like taggers and you could feel their fur! Sensory!
F3	Sensory!
F2	Board game.
F4	It's a board game like you say action figures?
All	[Multiple] Yeah. Yeah.
F2	And the action figure will each action figure will have a certain set of
F2	characteristics such as [cross talk]
F3	Yeah. Probably. Or something along those lines I suppose. I don't know what the exact rules would be
F3	but there should be some rules you know Have the kids think about the challenge

Figure 1: A screenshot of VizScribe shows representations of video, transcript, sketching activity, speech participation, and physical activity based on corresponding data. These data are displayed as interactive timeline views, such as video progress, transcript visualization, and sketch timeline. Word cloud and transcript views also allow the user to interact directly with the data. The user can jump to a specific part of the video by clicking the transcript, and can select a line to assign it a specific code. A short demonstration of VizScribe can be seen here: `https://vimeo.com/169905057`

4.1.1. Temporal View

Video data forms the central reference of this view, providing context to any of the other forms of temporal data. Visual representations of time-sequenced data should be designed to answer the question of "what was happening when. . . ?", by providing a means to align all temporal data such that it provides context to the analyst. The temporal view pane thus includes a video playback interface, with time encoded as a progress bar spanning the width of the pane. All timeline data visualizations are stacked under the video interface. They are scaled to, and aligned with the video progress bar, providing spatial coherence (requirement **R1**).

The temporal view features an event-level interpretation of protocols, including a timeline representation of the transcript. Our visual encoding of these events is


Timeline views for speaker F4
Verbalizations by F4 (transcript timeline)
Sketches created by F4 (sketching activity timeline)
Sociometric sensor-measured speech participation by F4 (speech participation timeline)
Sociometric sensor-measured physical activity of F4 (activity timeline)
video progress indicators
21:15 / 57:03
Graphical View of Transcript [click to contract view]
sketch Participation Chart [click to contract view]
Speech Participation Chart [click to contract view]
Activity Level Chart [click to contract view]
F3
F2
F1

F2: Like we can shoot Kryptonite out of the cannon balls for getting Super Man but not getting Luthor.

F2	Yeah. We're tryi
F2	So what I said t
F2	They can just sr
F2	And the child ca
F2	they have a bas
F2	Various modalit
F1	Yeah. And imag
F3	Yeah. For the c
F3	board game or
F2	Oh Like the one
F2	I mean there ar
F3	No. No. No. Th
F1	Yeah yeah yeal
F3	I'm talking abou
F2	Okay
F3	But it'd still have
F2	Good idea.
F3	Yeah.
F2	We can make a
F3	Yeah.
F2	having [inaudib
F3	Yes something
F3	Board game.
F3	And what if they
F3	Sensory!
F2	Board game.
F4	It's a board gam
All	[Multiple] Yeah.
F2	And the action f
F2	characteristics s
F3	Yeah. Probably
F3	but there shoulc

Figure 2: Detail of VizScribe's temporal view. The video used in this example has four participants or speakers, and thus each participant's activities are color-coded and displayed on the timeline. In this figure, all activities of participant F4 are illustrated as an example. The timeline of verbalizations shows a series of markers (🟨) indicating instances of F4's speech. Similarly colored markers are used for the remaining timeline views for F4.

inspired partly from temporal sequencing visualizations by Isenberg et al. (2008). The transcript is encoded as a series of thin vertical marks as shown in Figure 1, annotated as the "transcript" timeline visualizations. Each horizontal strip of marks represent utterances of one speaker, identified in the transcript file. Multiple speakers or subjects are color-coded accordingly across visualizations based on Harrower & Brewer's (2003) qualitative scheme. While the example shown in this paper does not use a colorblindness-safe set of colors, Harrower & Brewer's scheme includes sets of colors that work for those with users who have difficulty distinguishing colors. The dynamic linking between different visualizations (discussed under the "Orient" interaction in Section 4.2) provides another layer of visual connection that works in addition to the linking by color. The color coding is consistent across timelines, as seen in the figure. In the data used for this figure, sketching is performed on digital tablets and shared among users, allowing sketching activity to be logged. The visualization of this sketching activity also indicates through connecting lines the modification or extension of previous sketches, inspired from Robinson (2008) and Zhao et al. (2014).

This timeline visualization is also designed to be extensible (**R5**), wherein the design researcher, with some programming background, can modify existing timeline visualization panels, or add new panels of their own to suit their requirements.

For example, the speech participation data and accelerometer data collected from personal sociometric devices (Olguín et al., 2009) on each subject is visualized in the interface shown in Figure 1. This extensibility is explained in detail in Section 5. Finally, any coding performed on the transcript is also reflected in the timeline view, color-coded according to the user-defined codes. All of the above timeline visualizations are overlaid with a video progress indicator (Figure 1), visually linking them to the current time in the video.

4.1.2. Transcript View

The transcript view shows an interactive view of the transcript with corresponding text visualizations and annotations. Color-coded speaker identifiers provide a visual link to the corresponding timeline visualizations. In addition, within the transcript view, a *word cloud* view is generated from the complete text of the transcript after stop word removal. The words are scaled in proportion to the frequency of their occurrence in the text, helping identify recurring terms in the transcript. The transcript timeline view, text view, and word cloud together present multiple scales of representation (**R2**) to view participant utterances. Finally, a "code view" section within the transcript view allows the researcher to define and edit codes. A hierarchical code definition is enabled through tabbed entries in the field, thus supporting open coding. The code view is color-coded as well, and any transcript assigned to a particular code can be identified through these colors, thus providing context awareness (**R3**).

Each visualization embedded in these views is linked to related visualizations, shown implicitly through color and position, and identified explicitly through user interactions. These interactions are described in detail in the following section.

4.2. User Interactions

Fluidity of interaction and directness with which the user (in the context of VizScribe, the user is the analyst or researcher) can manipulate the data representations separate an effective visual analytics system from an ineffective one. Guidelines for designing such interactions have been determined with respect to mapping user intent (Yi et al., 2007) as well as fluidity of interactions (Elmqvist et al., 2011). The interactions designed into VizScribe address requirement **R4**, i.e. enabling fluid user interactions with the displayed representations. The three kinds of interactions designed into VizScribe are as follows:

- *Orient:* This is the act of the user familiarizing themselves with the data representations without explicitly changing its state. We map such actions to *hover* events: the dwelling of the mouse pointer on any element of the visualization, interpreted as the act of “raising the attention of the computer as a dialogue partner” (Müller-Tomfelde, 2007, p.561).

- *Focus:* This is the act of the user focusing their attention on one element on the data representations. We map such actions to “selection” events (mouse click and associated interactions) that change the state of the visualization.

- *Code:* This is the act of the user defining a code—a category of behavior or utterance—and/or assigning that code to a part of the transcript.

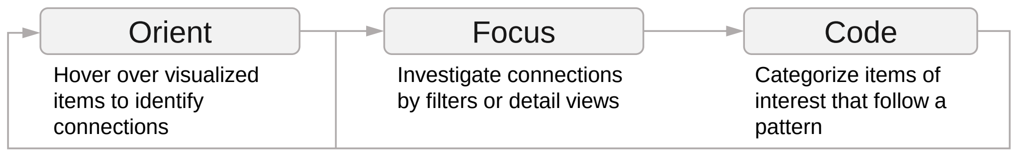

Orient
Hover over visualized items to identify connections
Focus
Investigate connections by filters or detail views
Code
Categorize items of interest that follow a pattern

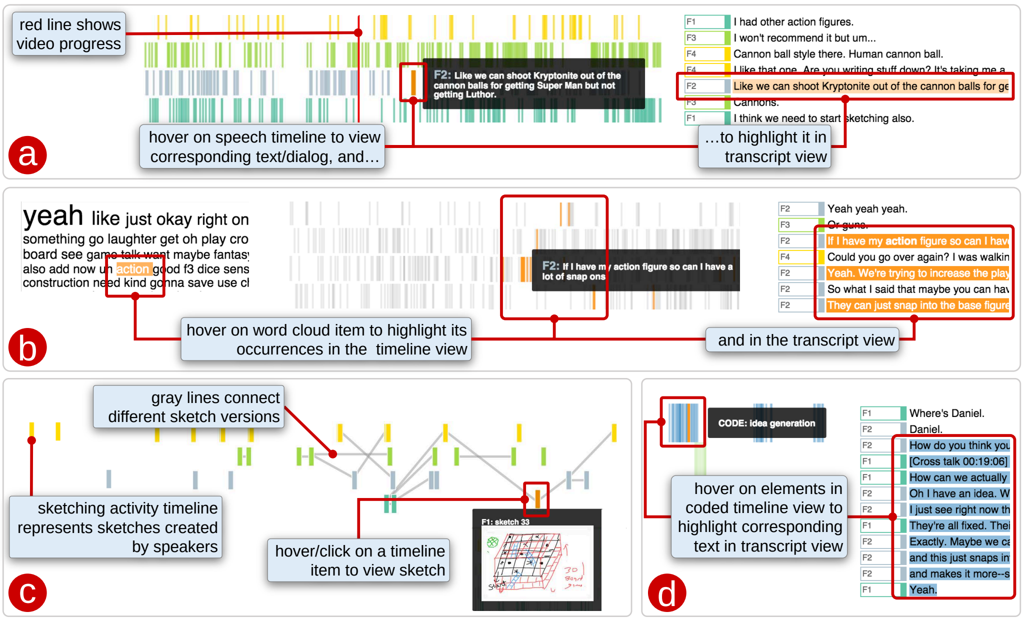
Figure 3: The flow of analysis designed into VizScribe. The user initially *orients* themselves to the various visualizations, using hover operations to explore connections. They then *focus* on items of interest, and finally begin *coding* these items into categories. The resulting coded visualizations are further explored to obtain better insight into the data.

The user’s process of analyzing the visualized datasets involves moving from one of the above acts to the other, as depicted in Figure 3.

4.2.1. Orient

Figure 4 shows some of the ways in which users can orient themselves to the data. Hovering on a line of the transcript highlights the corresponding mark in the timeline representation of the transcript, and vice versa. Similarly, hovering on a mark in the coded timeline view also highlights the corresponding line on the transcript. Hovering on a word in the word cloud highlights all the lines on which the word occurs, and shows a filtered view of these words in the timeline view (Figure 4B). Hovering on the sketch timeline view described in Section 4.1.1 display thumbnails of the corresponding sketches (Figure 4C).

These interactions provide a means to ensure that the user is not overwhelmed with the multiple data representations. They also allow the user a means of previewing points of interest in the dataset, before performing an overt interaction to *focus* on that point for a closer examination.


red line shows video progress
hover on speech timeline to view corresponding text/dialog, and…
…to highlight it in transcript view
hover on word cloud item to highlight its occurrences in the timeline view
and in the transcript view
gray lines connect different sketch versions
sketching activity timeline represents sketches created by speakers
hover/click on a timeline item to view sketch
F1: sketch 33
CODE: idea generation
hover on elements in coded timeline view to highlight corresponding text in transcript view
a
b
c
d

F2: Like we can shoot Kryptonite out of the cannon balls for getting Super Man but not getting Luthor.

F1 I had other action figures.
F3 I won't recommend it but um...
F4 Cannon ball style there. Human cannon ball.
F4 I like that one. Are you writing stuff down? It's taking me a
F2 Like we can shoot Kryptonite out of the cannon balls for ge
F3 Cannons.
F1 I think we need to start sketching also.

yeah like just okay right on
something go laughter get oh play cro
board see game talk want maybe fantas
also add now uh action good f3 dice sens
construction need kind gonna save use cl

F2: If I have my action figure so can I have a lot of snap ons

F2 Yeah yeah yeah.
F3 Or guns.
F2 If I have my action figure so can I hav
F4 Could you go over again? I was walkin
F2 Yeah. We're trying to increase the play
F2 So what I said that maybe you can hav
F2 They can just snap into the base figure

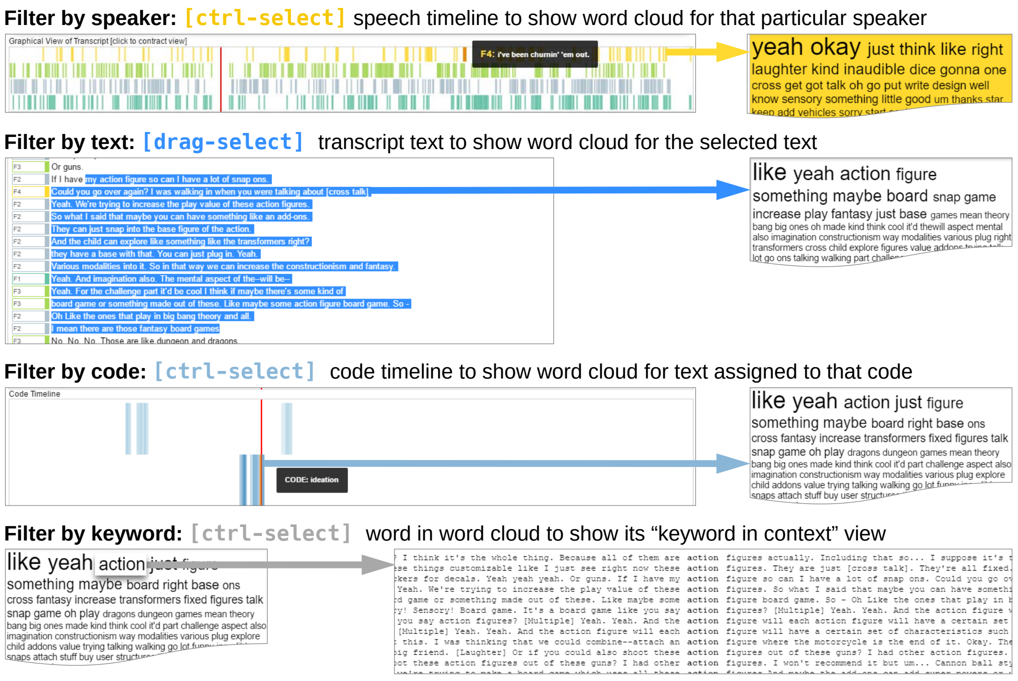
F1 Where's Daniel.
F2 Daniel.
F2 How do you think you
F1 [Cross talk 00:19:06]
F1 How can we actually
F2 Oh I have an idea. W
F2 I just see right now th
F1 They're all fixed. Thei
F2 Exactly. Maybe we ca
F2 and this just snaps in
F2 and makes it more--s
F1 Yeah.

Figure 4: Various forms of brushing and linking used in the framework, to facilitate the envisioned *orient* and *focus* tasks. (a) shows the transcript text on the right, linked to a time-sequence representation of utterance "events" color-coded by speaker ID. Hovering on an element in the time-sequence view highlights the corresponding text in the transcript view, and vice versa. (b) shows similar interactive linking, but this time, hovering on a word in the word cloud shows all its occurrences in the time-sequence and text views. (c) shows the sketch log view, where each "event" represents a sketch save operation. A hover event on this view shows a thumbnail of the saved sketch. The interactions in (d) are similar to (a) except the time-sequence view shows all utterance events assigned to a particular user-defined code.

4.2.2. Focus

In protocol studies, context is one of the main relations that needs to be examined. Yi et al. (2007) describe the act of "selection" in infovis as marking a data item of interest to keep track of it. Selecting an item in the temporal view has the effect of connecting this element to the video timeline, by skipping the video to the timestamp corresponding to the visualized element, to provide context. The timeline indicator on all other temporal views also skip to this timestamp, providing a visual cue of other temporal co-occurrences. Correspondingly, the transcript view scrolls to a participant utterance closest to this time, highlighting what was said close to the instant of interest.

In the transcript view, selecting any word from the word cloud persistently


Filter by speaker: [ctrl-select] speech timeline to show word cloud for that particular speaker
Graphical View of Transcript [click to contract view]
F4: i've been churnin' 'em out.
yeah okay just think like right laughter kind inaudible dice gonna one cross get got talk oh go put write design well know sensory something little good um thanks star keep add vehicles sorry start
Filter by text: [drag-select] transcript text to show word cloud for the selected text
like yeah action figure something maybe board snap game increase play fantasy just base games mean theory bang big ones oh made kind think cool it'd thewill aspect mental also imagination constructionism way modalities various plug right transformers cross child explore figures value addons
Filter by code: [ctrl-select] code timeline to show word cloud for text assigned to that code
Code Timeline
CODE: ideation
like yeah action just figure something maybe board right base ons cross fantasy increase transformers fixed figures talk snap game oh play dragons dungeon games mean theory bang big ones made kind think cool it'd part challenge aspect also imagination constructionism way modalities various plug explore child addons value trying talking walking go lot funny snaps attach stuff buy user structure
Filter by keyword: [ctrl-select] word in word cloud to show its "keyword in context" view
like yeah action just figure something maybe board right base ons cross fantasy increase transformers fixed figures talk snap game oh play dragons dungeon games mean theory bang big ones made kind think cool it'd part challenge aspect also imagination constructionism way modalities various plug explore child addons value trying talking walking go lot funny snaps attach stuff buy user structure

F3	Or guns.
F2	If I have my action figure so can I have a lot of snap ons.
F4	Could you go over again? I was walking in when you were talking about [cross talk]
F2	Yeah. We're trying to increase the play value of these action figures.
F2	So what I said that maybe you can have something like an add-ons.
F2	They can just snap into the base figure of the action.
F2	And the child can explore like something like the transformers right?
F2	they have a base with that. You can just plug in. Yeah.
F2	Various modalities into it. So in that way we can increase the constructionism and fantasy.
F1	Yeah. And imagination also. The mental aspect of the--will be--
F3	Yeah. For the challenge part it'd be cool I think if maybe there's some kind of
F3	board game or something made out of these. Like maybe some action figure board game. So -
F2	Oh Like the ones that play in big bang theory and all.
F2	I mean there are those fantasy board games
F3	No. No. No. Those are like dungeon and dragons.

```
? I think it's the whole thing. Because all of them are  action  figures actually. Including that so... I suppose it's t
ese things customizable like I just see right now these  action  figures. They are just [cross talk]. They're all fixed.
ckers for decals. Yeah yeah yeah. Or guns. If I have my  action  figure so can I have a lot of snap ons. Could you go ov
 Yeah. We're trying to increase the play value of these  action  figures. So what I said that maybe you can have somethi
rd game or something made out of these. Like maybe some  action  figure board game. So - Oh Like the ones that play in b
ry! Sensory! Board game. It's a board game like you say  action  figures? [Multiple] Yeah. Yeah. And the action figure w
 you say action figures? [Multiple] Yeah. Yeah. And the  action  figure will each action figure will have a certain set
 [Multiple] Yeah. Yeah. And the action figure will each  action  figure will have a certain set of characteristics such
t this. I was thinking that we could combine--attach an  action  figure where the motorcycle is the end of it. Okay. The
big friend. [Laughter] Or if you could also shoot these  action  figures out of these guns? I had other action figures.
oot these action figures out of these guns? I had other  action  figures. I won't recommend it but um... Cannon ball sty
```

Figure 5: Some of the filtering options available between the timeline and text views in VizScribe. The users can focus on specific parts of the datasets by filtering, say, the word cloud display by speaker or by the assigned code. They can also directly select a section of the transcript, which updates the word cloud to the selected text, providing an overview of the text. Finally, a keyword in context (KWIC) view of a word of interest provides a way to understand the context of its use in the transcript.

highlights all corresponding lines in the text and timeline view of the transcript. This allows the user to scroll around on the transcript, or skip to the timestamps of interest in the video to examine the data for patterns. Selecting a line on the transcript, when done through a keyboard combination, skips the video to the timestamp corresponding to that line, allowing the user to view any activity of interest occurring around the time. It also allows the user to check for any temporally co-occurring item of interest in the timeline visualizations.

Such interactions also allow for the *filtering* and *details-on-demand* tasks specified by Shneiderman (1996). Filtering is the removal or de-emphasis of uninteresting items, allowing users to focus on their interests. Details-on-demand refers to obtaining details of a particular selection, usually displayed on separate pop-up windows. In VizScribe, filtering is achieved in several ways: selecting a block of text in the transcript updates the word cloud to contain words from

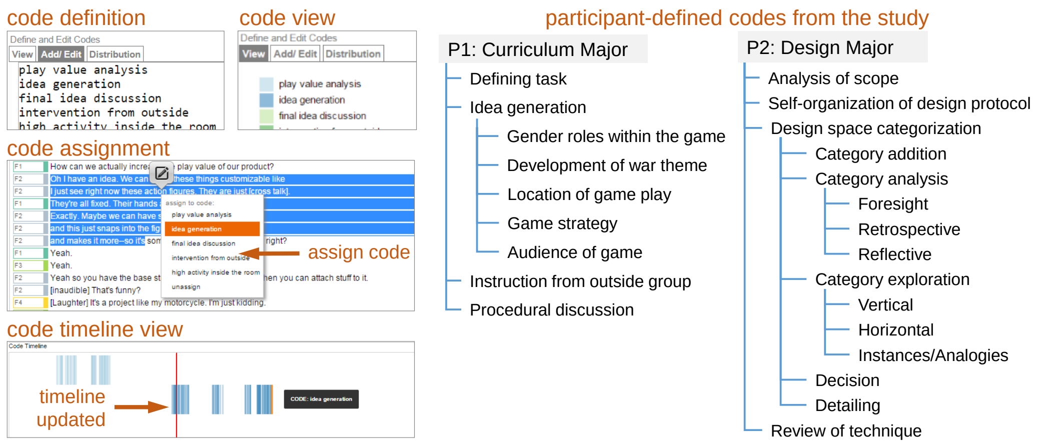


Figure 6: Coding in VizScribe and sample codes generated by participants from our formative studies. The screen captures from VizScribe on the left show the code definition and code assignment. The hierarchical codes on the right were defined by the participants (P1 & P2) in the formative study. Though using the same dataset for the coding task, the participants produced very diverse codes, based on their research backgrounds (curriculum development vs. design).

only the selected block. This filtering is extended to speakers and coded sections of the transcript: selecting a speaker (with a keyboard shortcut) on the transcript timeline, updates the word cloud to only show words spoken by that speaker, while selecting a code from the coded timeline view updates the word cloud in the same way. VizScribe has two forms of details-on-demand tasks: selecting a word in the word cloud (through a keyboard shortcut) generates a *keyword in context* view showing text immediately preceding and succeeding every occurrence of that word, allowing the user to identify patterns in the context of utterance of that word. These and other filtering interactions are shown in Figure 5. A similar selection of an item in the sketch activity timeline shows a larger view of the sketch in a separate window, allowing the user to examine its details.

4.2.3. Code

Typically, this data exploration is followed by the categorization of utterances and behavior, and coding or annotating these with the identified categories. VizScribe is provided with a text-entry field for determining such code. Hierarchies of code are encoded by the user through the use of tabbed indentations in the text entry. Once a code is defined, they are assigned to a selected block of text through a context menu. A unique color for each code allows the coded transcript text to be highlighted on demand, as shown in Figure 4(d). At any stage, the coded transcript can be exported as a comma-separated value (CSV) file for use with other tools. Figure 6 shows the coding process and sample codes created by two participants in our formative studies. The diversity of the code generated and the hierarchies formed show how VizScribe supports open, axial coding practices.

5. Framework Implementation

The introduction of visualization libraries such as D3.js (Bostock et al., 2011) has resulted in the web browser emerging as an appropriate platform for infovis, for both collaboration and dissemination. We chose to implement VizScribe as a web-based framework, with the data uploaded and stored at the server, and then processed and visualized at the client end. This implementation has the added advantages of platform independence, minimal installation requirements, and for future extension into a collaborative visual analytics framework.

The application is hosted from a Node.js[5] server. Design activity data can be uploaded to this server by an analyst using VizScribe to create a coding session. In the current interface, coding sessions are automatically saved at the server (in the form of data logs), but are not retrievable by the framework, so support for multiple sessions and multiple users is yet to be implemented. The data is cached and hosted for all client web browsers running the VizScribe interface. Almost all data processing into structural and graphical elements for the visualizations occurs at the client end. All visualization elements in VizScribe have been developed using HTML5 and JavaScript, with the visualizations generated primarily using the D3 library.

From the server, the video is streamed, on demand, to an open-source HTML5 video player called Video.js[6]. The interface thus never stores the entire video but only chunks of it at the current playhead, which can be paused, reversed, or fast forwarded. This protocol is therefore scalable to large video sizes and was handled through websockets, specifically by using the socket.io[7] library.

The data processing pipeline is shown in Figure 7. Input data is first uploaded to the server in the form of video and a transcript file, with additional time-stamped data including, but not restricted to sketch logs and corresponding image files of sketches, instrumentation logs and activity logs. While it is essential that the transcript file includes timestamps, speaker identifiers for each line of spoken

[5]http://nodejs.org/
[6]http://videojs.com
[7]http://socket.io

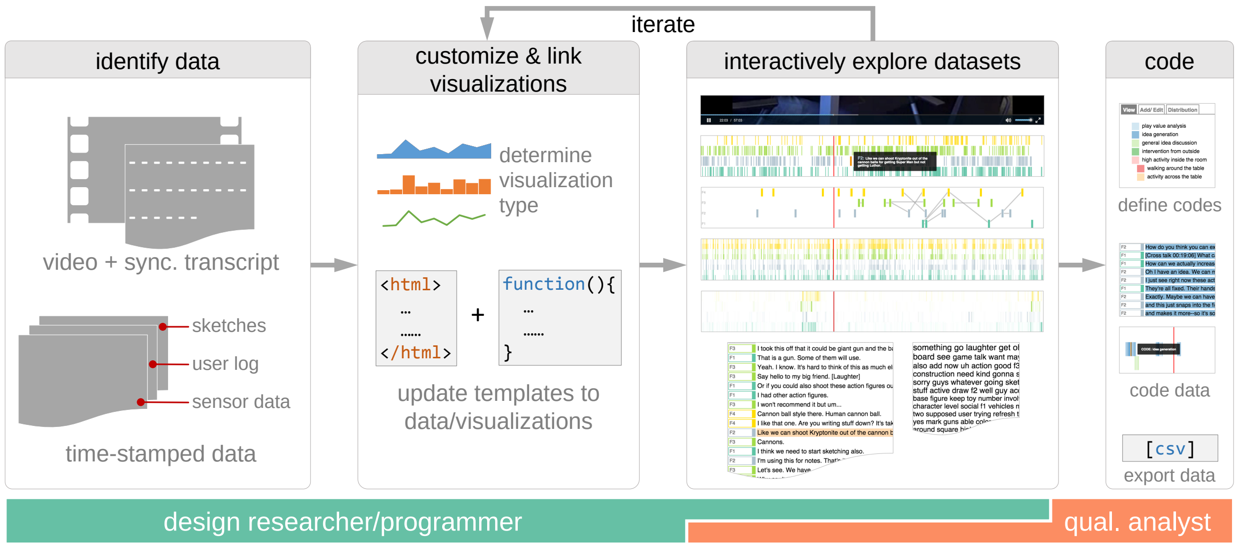


Figure 7: Visualization pipeline used in the VizScribe framework. The standard inputs (required) are in the form of a video and a timestamped transcript. The VizScribe web application generates a default timeline and word cloud visualization for the transcript. Predefined code "templates" cater to other timestamped code, where the researcher, based on step-by-step instructions in the VizScribe Wiki, can iteratively customize these templates to explore different interactive visualization forms. Once the visualizations are finalized, the researcher can begin coding the transcript and exporting the coded data to a comma-separated value (csv) file.

text are optional. The current data import requires transcripts to be in a comma-separated value (.CSV) format, but is extensible to include other formats such as the SubRip (.SRT) format or closed captioning formats.

All log files, including sketching activity logs, need to have timestamped entries, and corresponding references to additional media where applicable, such as sketches or images. These additional media files, which can include images, vector graphics, and notes, are uploaded as a single archive. This upload mechanism, showcased with sketches in Figure 7, is extensible to the other media types.

The transcript is reformatted into objects as specified under the *document object model* (DOM), ensuring cross-browser compatibility. In addition, this allows each object of the DOM to be programmatically accessed. VizScribe uses the JQuery library[8] to easily access and manipulate DOM elements, and implement the interactive linking between objects as discussed in the previous sections.

[8]http://jquery.com

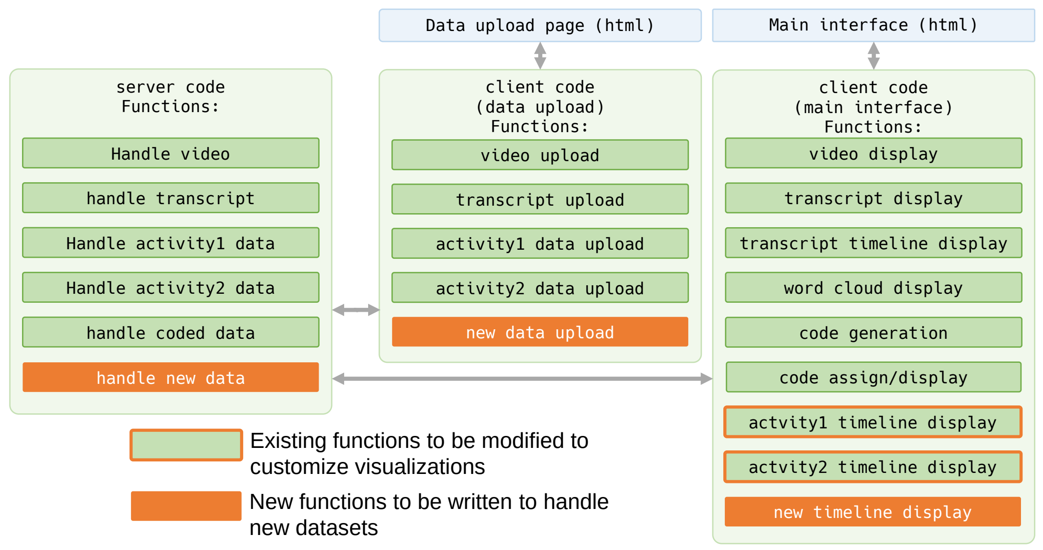


Figure 8: An overview of the high-level functions in VizScribe, showing the functions that need to be modified in order to edit or customize existing timeline visualizations, and functions that need to be added in order to create new timeline visualizations. Figure 9 shows conceptually how these visualizations can be customized. Templates for functions are also available in the VizScribe source code page (see footnote 8) with detailed instructions on its Wiki.

5.1. *Customizing and Extending VizScribe*

Timeline entities are parsed and mapped to attributes of D3 graphical objects, aligned with the video timeline. We take advantage of D3's *data* operator, a format-agnostic array of values that are linked to visual elements displayed on the screen. Figure 9 shows how a timeline dataset is mapped on to a D3 object. This is also the means behind VizScribe's extensibility: new time-series datasets can be represented in an appropriate visualization by adapting any of the existing time-series visualization classes, or making use of code templates made available along with the source code[9], with detailed instructions on customizing and extending VizScribe. While the step-by-step instructions on extending VizScribe lay beyond the scope of this paper, and are available in the Wiki accompanying the source code, this section will provide a brief overview of the process.

Figure 8 shows the high-level functions in VizScribe's code, highlighting the functions that need to be modified to customize existing timeline visualizations, and the new functions that need to be added in order to extend VizScribe to create

[9]`https://github.com/senthilchandrasegaran/vizscribe`

new timeline visualizations.

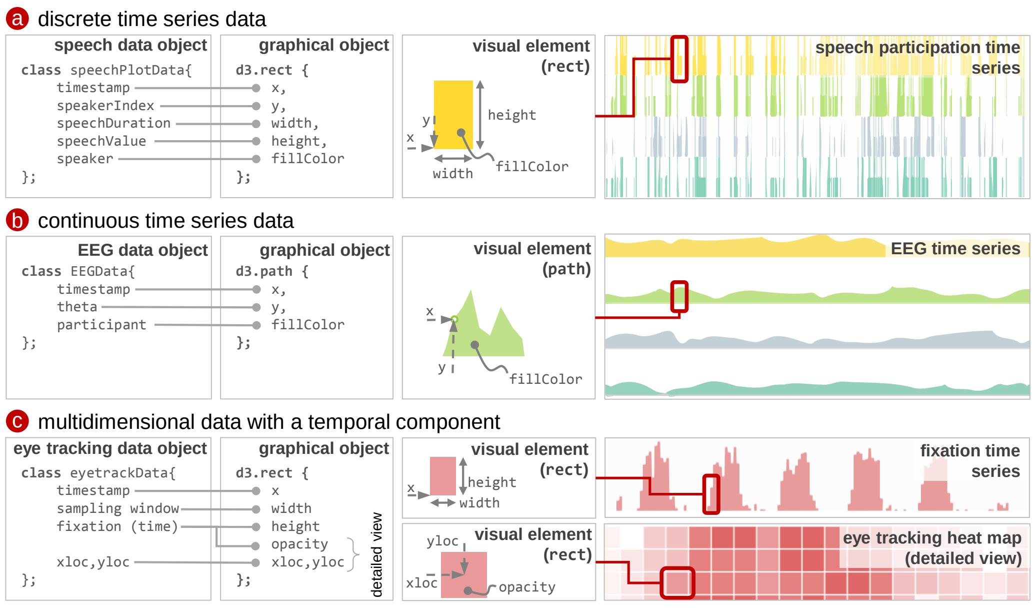
Protocol studies are no longer just about video recordings and transcripts. The use of sensor data to study designers to better understand how they think, plan, interact, and make decisions, is not new. For instance, Göker (1997) used EEG sensors to correlate behavioral experiments with novice and expert designers and electrophysiological experiments using EEG sensors to find that novices use more of their reasoning (frontal lobe activity) to solve problems, while experts use visual recognition (parietal lobe activity) to draw on their experience to achieve the same. Similarly, Bi et al. (2015) use eye-tracking data to understand the forms of visual stimuli that affect design decision-making in test "game" scenarios. Finally, wearable sociometric sensors (Olguín et al., 2009) are being used to study social behaviors within groups, e.g. Gloor et al. (2012) or even provide real-time mediation of brainstorming sessions to ensure equal participation (Kim et al., 2008). We use these sociometric sensors in our user study.

To illustrate how VizScribe can be extended to include such data, Figure 9 shows how the timeline elements can be mapped to data elements in the case of (a) discrete time series data measured over intervals of time, such as speech participation measures, (b) continuous or streaming data such as EEG sensor readouts, and (c) multi-dimensional data that have temporal components, such as eye-tracking data or user behavior logs that require additional visualizations in addition to time-series visualizations. Standard event controllers are available that can be changed to determine actions to perform when a hover or select event occurs. Selecting an appropriate visualization is an iterative process. By experimenting with the various geometric entities available in D3, appropriate visualizations can be explored.

We envision VizScribe to be used by design researchers, who, with some programming background, will be able to adapt the visualizations to forms of data pertinent to their work. As mentioned earlier, VizScribe's source code page has templates for the required functions, and a detailed Wiki with instructions on installation, use, customization, and extension of the framework.

6. Framework Evaluation: Overview and Rationale

The goal of VizScribe is to support protocol analysis using a visual analytics approach by providing interactive visualizations that provide multiple modes of exploring the data. Given this focus, we sought to first understand the utility of the visualizations when the analysts are given a low-level coding task. Our formative studies thus focused on relatively lower-level and smaller-scale coding tasks that



```
class speechPlotData{
    timestamp
    speakerIndex
    speechDuration
    speechValue
    speaker
};
```

```
d3.rect {
    x,
    y,
    width,
    height,
    fillColor
};
```

```
class EEGData{
    timestamp
    theta
    participant
};
```

```
d3.path {
    x,
    y,
    fillColor
};
```

```
class eyetrackData{
    timestamp
    sampling window
    fixation (time)

    xloc,yloc
};
```

```
d3.rect {
    x
    width
    height
    opacity
    xloc,yloc
};
```

Figure 9: Extending or customizing a timeline visualization to fit custom time-series data involves mapping attributes of that data to a corresponding visual object. VizScribe uses D3 for this purpose, whose data structure makes this mapping possible. The above figure illustrates how this extension is possible for three main categories of data, namely a discrete time series data where data is sampled at intervals, b continuous time series data where data is read in a stream, and c multidimensional data with a temporal component, where a time series visualization needs to be augmented with additional visualizations. For all three categories, the above figure illustrates ways in which the VizScribe timeline views can be extended to incorporate such data by mapping data attributes to geometric attributes of appropriately chosen D3 elements. Hover/click behaviors are then specified, allowing for interaction with the data.

would help us improve the framework. We followed this up with a summative study that contained a more diverse set of tasks for a full-fledged dataset. This section provides an overview and justification of the studies performed, while the next sections detail the methods, participants, and findings from these user studies. The sequence of studies is outlined below:

- **Formative Study**: This study sought to identify weaknesses in the design and implementation of VizScribe, which helped us separate useful visual representations and interactions from redundant ones, and enhance those that showed promise. We performed two formative studies:
  (a) *Prescribed coding*, to assess how participants navigated the framework given a specific coding task (see Section 7.1); and

(b) *Open coding* to assess the ability of the framework to support analysts with diverse analysis goals (see Section 7.2).

- **Framework Redesign:** Based on the findings from the formative studies, we implemented changes to the framework (see Section 8).
- **Summative Study:** This helped us understand the versatility of the interface in answering certain closed- and open-ended questions of the kind that interest design researchers (see Section 9).

For each study, a brief demonstration of VizScribe and its features was provided to the participants, after which they were allowed as much time as they needed to familiarize themselves with the interface. On average, participants took approximately 60 minutes to complete the tasks in the formative evaluation, and 75 minutes in the case of the summative evaluation.

6.1. Context: Data from Design Sessions

Given our goal to understand design processes through a variety of data forms, we recorded two student teams each comprising four members) working on a design task as part of a mechanical engineering graduate course on product design. The teams were recorded when working on a design modification assignment that required the team to first categorize a given toy according to its "play value" (Kudrowitz & Wallace, 2010), and then modify it to extend and/or change the play value. The teams used skWiki (Zhao et al., 2014)—available as open source[10]—to create, exchange, and modify sketches of their ideas, enabling us to log all sketching activity during the design session. We created a timeline visualization of the sketching activity from the skWiki sketch log data as shown in Figure 4(c). Both the design sessions were also video recorded with the consent of the participants. One team was also fitted with wearable sociometric sensors (Olguín et al., 2009; Kim et al., 2012) to record their speech patterns and body movements (Fig. 10).

We used a 15-minute video segment and sketch data from the team *without* sociometric sensors for the formative evaluation, and the complete 60-minute segment and associated data from the team *with* sociometric sensors for the summative evaluation. This decision was in accordance with the goals of the evaluations, with the formative evaluation focusing on studying the efficacy of providing temporal

[10]`https://github.com/karthikbadam/skWiki`

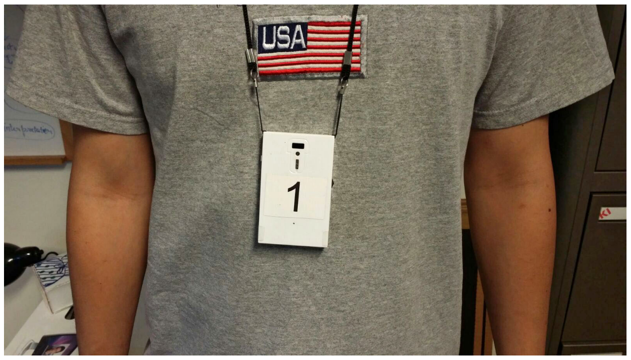


Figure 10: The wearable sociometric badge used in the design session. The badge, developed by Sociometric Solutions (now Humanyze: `http://humanyze.com`), measures individual biometric data including speech events and body movement, as well as social interactional data such as face-to-face interactions, conversational overlaps, turn-taking, and proximal connections.

and transcript views for coding, and the summative evaluation for understanding high-level user behavior patterns using multiple and integrated forms of data.

In both studies, VizScribe was run on a Google Chrome browser and displayed on a 20-inch LCD screen. The data was uploaded and rendered on VizScribe before the start of the experiment.

6.2. A Note on the Study Rationale

As we have previously noted, VizScribe represents a visual analytics *approach* to support the analysis of data provided by protocol studies, and not a CAQDAS tool unto itself. This distinction is important, as VizScribe is an exploration of the techniques existing CAQDAS tools can use in order to better support multimodal data analysis.

Our evaluation of the framework thus focuses on the utility of the visualizations and interactions designed with the "orient–focus–code" paradigm in mind, first in parts in the formative study, and then as a whole in the summative study. We chose to not conduct a comparative study with existing CAQDAS tools as it would be counterproductive to the goal of this work: On the one hand, existing CAQDAS tools do not use visual analytics approaches, while on the other hand, VizScribe does not provide the full-fledged protocol analysis and coding support that CAQDAS tools do.

For example, datasets such as sketching logs, or activity data from a body-mounted sensor, are very difficult, if not impossible, to meaningfully analyze using NVivo or other CAQDAS tools. While most CAQDAS tools include features to integrate multiple datasets, their focus is on consolidation rather than visual coherence. This also applies for research prototypes such as Chronoviz, which focuses on event/activity timelines. For example, the ability to filter the video timeline through word occurrences in order to, say, *focus* on all the instances of the team discussing a particular concept, is a function that is not available in such tools, and is therefore not typically used during protocol analysis. While the advantage of an effective visual analytics-based design of CAQDAS tools may be illustrated through a comparative study, it would be premature for us to attempt it, when our focus is to determine what a visual analytics-based approach to protocol studies entails. We thus focused on evaluating VizScribe's usefulness in qualitative analyses, particularly the effectiveness of the interactive data visualizations.

7. Formative Studies

We conducted formative studies to evaluate the use of the interface in performing specific tasks in data navigation, and to understand the effectiveness of the framework in performing open-ended exploration and coding of the data.

We first used a prescribed coding scheme, requiring participants to identify certain kinds of behavioral interaction in the design teams during a 15-minute segment of a team brainstorming session. In the second study, participants were given a freer rein to code interactions or behaviors that they found interesting, following an open coding scheme. Our goal was to examine how VizScribe was used by analysts with different analytic goals, and to identify functions that could be changed, removed, or added to the framework.

7.1. Study I: Prescribed Coding

Closer in nature to *selective coding* Corbin & Strauss (1990), we called this task *prescribed coding* as we provided a core set of behaviors that the participants were asked to code. We recruited 6 paid participants (5 male, 1 female), aged between 27 and 32 years. All were graduate students, three of whom were experienced in ethnographic analysis, and the remaining three experienced in design process analysis. Two of the participants had prior experience in protocol analysis.

Participants were given 60 minutes to familiarize themselves with the data shown in the VizScribe interface (video, transcript, and sketch logs). They then used the transcript data and coded instances of *idea generation*, when a subject in the video comes up with an idea for the toy design, *idea extension*, when a suggested idea is modified or developed, and *idea critiquing*, when feedback is provided to a suggested idea.

Following the coding session, participants answered a set of questions providing feedback on their experience with the interface. This included both open-ended questions on their overall experience and feedback, and their rating of the usefulness and ease of use of VizScribe's features on a five-point Likert scale.

7.1.1. Results

In general, participants liked the way multimodal data was presented, and found it easy to navigate in the interface. One participant summarized: *(VizScribe is a) "useful tool for observing multiple modes of data. I was able to relate the transcript to actual body language from the video and sketching activity. I liked how I needed minimal instructions to use the tool. A browser based environment really helped, as I was familiar with most browser based interactions."*

Among the participants, 100% reported that the interactive transcript was the most useful feature of VizScribe, while 66% reported that the word cloud was the least useful. This is partly explained by a lack of an appropriate motivation to use the word cloud: the visualization is more useful for an overview of a large body of text, and a transcript of a 15-minute conversation was small enough that the participants could simply read through it. In the temporal views, 83% of the participants reported the sketch timeline was intuitive but not relevant, partly because this view was not linked to other visualizations, and partly because of the small number of sketches.

Finally, with regards to coding the transcript, participants expressed the need to assign multiple overlapping codes to the same sections of the transcript, and to to assign codes at a word-level granularity in the transcript, i.e. select a part of a line and assign a code to that part.

7.2. Study II: Open Coding

For the second study, to assess the breadth and flexibility of VizScribe, we recruited two paid participants (one male, one female), both graduate students aged 29 and 34 years. The first participant, a curriculum and instruction student in art education, had prior experience with coding in NVivo, and the other, a student of mechanical engineering majoring in design, had no prior experience in coding or protocol analysis. They were asked to use an inductive open coding technique (Glaser & Strauss, 2009)—which was first explained to the participants—to identify interesting actions, processes, or behavior, code such events, and categorize them hierarchically by simply indenting the respective codes in the code editing text box. Participants had 60 minutes to perform this task, followed by an open-ended survey where they discussed the coding process they used and their feedback on VizScribe's features. Our goal was to evaluate the utility and ease of use of VizScribe in a real-life coding scenario, and identify challenges that arise in data representation and coding.

7.2.1. Results

Both participants found it easy and intuitive to create multiple code hierarchies. The code sets created by each were different, reflecting their different backgrounds and research orientations, which supports the goal of VizScribe providing support for grounded theory practice in open coding. One of the issues the particpants faced was the representation of codes on the transcript when numerous codes were created. The color-coded display seemed to become cognitively difficult to process when there were 10–12 colors, one for each code, painted over the transcript. Additionally, the transcript would retain the color of only the most recently-assigned code, thus hiding instances of multiple codes assigned to the same section of the transcript. By comparing the codes generated by both participants, we found that both viewed the same dataset for approximately the same duration, and yet created markedly different codes (Figure 6).

8. Framework Redesign

Results from formative studies helped identify features that were useful in exploring multimodal data, and features that needed refinement. Based on our observations and from the participant feedback, we made the following enhancements to the framework.

Scale-robust timeline view: The transcript timeline visualization became less readable when longer videos of around 60 minutes were used. The visualization was thus extended to indicate, through color codes and position, the different speakers in the visualization. A "magic lens" local zoom was added to the view for easier selection of a visualized element.

Text filters: The word cloud view was made more useful by introducing more techniques for "focus" tasks explained in section 4.2. This included the filtering tasks explained in the section, such as word clouds specific to one speaker's utterances, word clouds specific to certain coded parts of the transcript, or a dynamic word cloud that updates itself to a selected block of text.

Interactive code linking: The color limitation that was faced by using more than 12 codes was mitigated to an extent through enhanced interactions. The code colors on the transcript were made non-persistent, i.e. they "faded" two seconds after coding, so that transcript legibility was not compromised. By hovering or clicking on a code in the coded timeline view, users could reveal the corresponding color on the transcript. This ensured that the transcript was overlaid with only one code (color) at any point of time, making it visually less cluttered and circumventing the problem of overlapping codes identified in the open coding study.

Extensibility: The extensibility discussed in section 5 was developed as a result of participants suggesting more intuitive links between the sketch timeline and the rest of the time-sequence data, and suggesting other datasets that could be visualized and connected to the VizScribe timeline. We thus linked the sketch timeline to the video timeline to enable skipping to a particular time when a sketch was saved on the server by simply clicking on the corresponding timeline element. The extensible, interactive timeline views were developed as a generalization of the timeline views.

We conducted the summative study after implementing these changes to the framework.

9. Summative Study

Summative evaluations are useful when one needs to understand high-level behavioral trends, and mental models of participant behavior (Hix & Hartson, 1993). In this study, our goal was to observe how participants use the interactive visualizations of multimodal data to develop an *awareness* of events in the design session, as well as an *interpretation* of the activities, roles, and processes revealed by the data. This helped us understand the different means by which analysts approach design protocol studies, and the versatility of the visualizations used in VizScribe to cater to these means.

9.1. Participants and Procedure

We recruited ten paid participants for this study (5 female, 5 male), all between 23 and 36 years of age. Six were PhD design majors in mechanical engineering, two were engineering education majors (one master's and one PhD), one was a master's student in computer information technology with a background in teaching undergraduate design courses, and one was a post-doctoral researcher in educational psychology, focusing on design education. Nine of these participants were familiar with qualitative analysis, ranging from document analysis, to interview coding, to analyzing videos of participant gestures, and one had prior experience with CAQDAS tools (NVivo and Dedoose). Four of the participants had used visual analytics tools such as Jigsaw (Stasko et al., 2008).

All participants used VizScribe to complete a set of seven tasks to analyze a 60-minute design session video. In addition to the video, transcript, and sketching data in this study, the data visualizations also included sociometric sensor data that captured the *activity levels* (from accelerometer sensors), and *speech participation* (speaking to listening ratios) of each participant. The seven tasks to be completed were:

T1 Identify the sketch/sketches that represent the *final idea* chosen by the team.

T2 Identify the team member that originated the idea that eventually evolved into this final idea, and code the part of the transcript that refers to the first mention of this idea.

T3 Did the team discuss other ideas before they narrowed down on one final idea? If so, describe these ideas.

T4 Code the parts of the transcript that you identify as the start and end of the team's divergent process (ideation) and the start and end of the team's convergent process (evaluation and selection).

T5 Using the word cloud, identify themes in each team member's utterances.

T6 Identify and code three instances with no speech overlap between team members, and three instances where there is speech overlap. What behavior differences can you observe between the two categories?

T7 Using the activity timeline, identify with timestamps three unique instances of activity or movement, and describe them.

Tasks T1 and T2 have components that can have "correct answers" or answers that are comparatively less open to interpretation. We used these tasks as "ground truth" tests, to determine if the VizScribe interface helps the participants glean such information. Tasks T3 through T7 were open-ended questions designed to assess how VizScribe is useful in understanding aspects of the presented data that are not immediately apparent from the video and the transcript alone. T3 and T4 were chosen to verify whether the additional data representations were useful when the task would otherwise involve sifting back and forth through the transcript and video. For instance, the sketch timeline along with the transcript could be relevant to T3, whereas the interactive word cloud could be used to filter the transcript when attempting T4. T5 was intended to test if the word cloud could provide useful overviews, and T6 and T7 were included to draw the participants' attention to the sociometric sensor data to determine the utility of the data and the visualizations.

We asked participants to complete surveys at multiple points during the data analysis session, so that we could provide probes pertaining to specific tasks immediately after the task was executed. The final survey asked participants about the utility of specific features such as the coordinated data representations, and the interactive transcript display, as well as an overall rating of VizScribe using the System Usability Scale (SUS) (Brooke, 1996).

9.2. Results and Discussion

In analyzing both participant feedback and the logs of their interactions with the different features of VizScribe, we found a few patterns in the features of VizScribe that they used for certain kinds of tasks. We found that they used the filtering aspect of the word cloud to search or navigate the dataset for specific answers. On the other hand, for the more interpretive tasks, participants tended to use different features for the same tasks, illustrating a richness that VizScribe affords in visualizing data. Finally, the alternate, nonverbal timeline visualizations, provided a new lens for participants with which to view the design process, and form new associations between kinds of behavior. These and other findings are discussed in this section.

Table 1: Mean task completion times with standard deviations.

	Task	**Duration** (min)	
		Mean	*SD*
T1	Identify final idea	06:45	*03:42*
T2	Identify originator of final idea	09:07	*03:27*
T3	Describe other ideas discussed	11:14	*04:55*
T4	Identify start & end of ideation & selection	19:20	*08:27*
T5	Identify themes for each team member	09:08	*03:47*
T6	Describe 3 instances with & without speech overlap	12:33	*04:58*
T7	Describe 3 unique activity instances	05:58	*02:45*

Participants spent an average of 75 minutes for all tasks combined, but there was considerable variation between participants across the tasks. On average, T3, T4, and T6 were the most time-consuming tasks. T3 and T4 were expected to take more time as they required exploration of almost the entire design process using the speech and text timeline. T6 required the use of speech overlap data from the sociometric sensors, and the high number of the overlap events made it difficult for

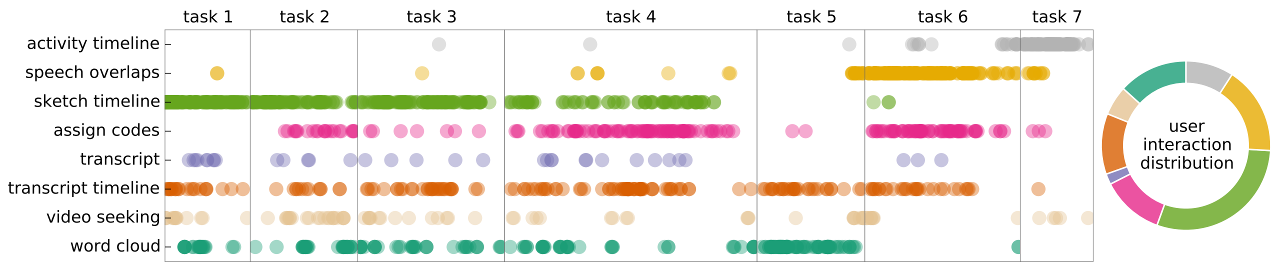


Figure 11: Task-wise usage of the different elements of VizScribe, aggregated over all participants. Views such as the word cloud, the transcript timeline, and the sketch timeline can be deemed the most versatile, since they are used across most of the tasks. The distribution shows that most of the exploration of the provided dataset occurs through the "sketch timeline": the view showing the creation and development of every sketch by the subjects in the video.

the participants to sift through the view. Table 1 presents the mean task times for each of the seven tasks.

Examining participant performance in these tasks, we found that among the "closed-ended" tasks, all ten participants correctly identified the sketch chosen by the team (T1), and 80% of the participants correctly identified the author of the corresponding idea (T2). Even the two who answered differently were not necessarily wrong: they chose the participant who had *sketched* the idea, and not the member who *originated* the idea. The consistency of the answers to these tasks attests to the veracity and legibility of the data representations.

From the analysis of the participant activity logs, we can see that the transcript timeline and the sketch timeline were the most used, especially in tasks T1 through T4. These were tasks that required the participants to understand the process followed in the recorded design session (Figure 11). The word cloud, however, was used sparingly for this analysis. In design protocol analysis, sketches and verbalizations are the chief ways in which designers externalize their thoughts, and it thus follows that interactive timeline visualizations of this data would be the most used to understand the process. Tasks T5, T6, and T7 were designed to utilize the word cloud and the speech participation and activity timelines respectively.

The most interesting interactions of the participants involved filtering and navigating text views, using different visualizations for the same task, and observations of the video-recorded behavior by interacting with activity sensors, which we discuss briefly below:

Filtered navigation: Participants used the word cloud as a filter for the transcript view, especially when performing a focused search for a specific idea generated by the design team. In completing T2, for example, after identifying the final product was a board game, the participants used the word cloud and its link with the transcript (text and timeline) to filter for all occurrences of the word "board". One participant explains: *"there were many cases where I saw an interesting pattern on the timeline, watched the video, read the transcript, found a word, and then saw where else it occurred in order to see if there was a pattern."* By focusing on these filtered sections of the transcript, participants were able to identify the right instance at which the idea was first discussed. Task T5 explicitly required the use of the word cloud, where participants were to identify themes from each video subject's utterances, preferably using the *"filter by speaker"* technique explained in Figure 5. However, in addition to this, three participants tagged the same subject as the leader of the team captured in the video—an unexpected and interesting use of the word cloud to determine subject role/personality. This form of subject characterizations through a summary view, while convenient, can be antithetical to the rigor of good coding practice. This is the space that the VizScribe framework occupies: it allows the analyst to skim through a larger dataset, identify questions of interest, and focus on these questions while examining the details.

Multiple approaches to explore datasets: We also observed that the multiple views of the data allowed participants to use diverse analytic approaches: different participants oriented themselves differently in the dataset, and thus picked different "entry points" – most notably, the transcript timeline, sketch timeline, and transcript text views (see Figure 11). This makes sense: conceptual design predominantly involves sketches and explanations of these sketches, when working in a team.

This use of diverse entry points has the potential to support broader research questions, accommodating the analyst in choosing from a variety of paths from which to approach the questions. In the example above, providing representations of both speech and sketch activities to the analysts allows them the flexibility of choosing the data representation based on the research question being asked. It would thus seem that it is not just the dataset that determines the most useful representation, but also the *user* who determines it. While this may seem obvious, this insight is often missing from most CAQDAS tools: there are few alternate representations of the same dataset.

Temporal correlation of multiple data streams: Task T7 required participants to pay attention to the activity timeline view, and note unique instances and corresponding subject activity in the video. This was done to raise the participants' awareness of the usefulness of such representations. Sure enough, participants found that they could not only make note of larger events such as subjects leaving the room, but also smaller instances of them leaning across a table, or even

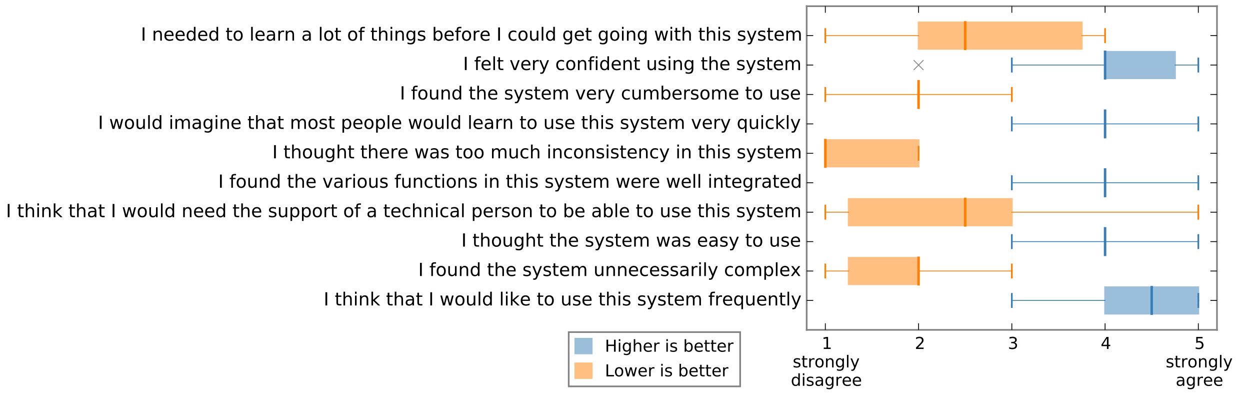


Figure 12: The System Usability Study (SUS) scores shown category-wise, aggregated over all participants. Plots in orange are better when lower, while plots in blue are better when higher. An mean rating of 75.5/100 was obtained.

picking up a toy from the table. One participant remarked *“traditionally, it has been difficult to track down the correlation between verbal behavior with gestures, sketching behaviors. With the help of these features, researchers would have a much easier time to pinpoint the relations between these behaviors.”* Interactive visualizations of such data thus provides an opportunity to assess both macro-level behaviors such as entering and exiting spaces, as well as micro-level behaviors such as gestures and object manipulation.

The framework usability received a mean rating 75.5/100 on the SUS, which is “*acceptable*” on the acceptability range and “*good*” on the adjective ratings (Bangor et al., 2009). Figure 12 illustrates the response distribution for the SUS. The average SUS rating by participants with a prior background in infovis was 79.4, while the rating by participants without this background was 72.9. While there is a fair difference between the two group ratings, both scores fall within the same acceptability and adjective ranges as the overall mean rating. The difference between the groups is understandable given the short exposure the participants had to the interface: participants more familiar with interactive visualizations would be expected to adapt quicker to the interface. A more accurate measure of the usability of this system will need a longer-term study, which is outside the scope of this work. Participants’ subjective feedback was largely positive, with the best summing-up provided by one participant who had worked with CAQDAS tools (but had no background in infovis): *“The tool is a great synthesis of what other tools have been missing, as I have used them.”*

10. Implications for Qualitative Analysis

The process of qualitative data analysis is aptly described by Creswell (2012, p. 182) as a "data analysis spiral." This spiral represents organization and management of data, reading, reflecting, annotating the data, identifying and comparing contexts and categories, interpreting and describing, as well as representing and visualizing the data. Data analysis is an iterative process, involving alternating deep dives into the data, perusing and annotating to get a sense of the data as a whole, examining it in parts to identify patterns of interest, as well as winnowing it to select information of relevance. Drisko (2013) emphasizes how qualitative data analysis software has made possible research endeavors into multimedia data, by allowing the display of images, audio, and video, and the relationships between data that can be inferred directly on images or indirectly on video/audio timelines.

A visual analytics approach such as ours allows qualitative researchers to visualize and interpret design behaviors through new data forms that are linked to traditional text and video data. These data provide the grounding for "thick" descriptions of design behavior (Geertz, 1973) that provide contextualized explanations. For example, an unusually high level of movement activity on the timeline can be directly queried to view in the video the movement of the design team into or out of the room, or to a display wall to collaboratively sketch or discuss a design. The sketch timeline can reveal, at the same time, the design under discussion. Selecting the transcript around this time updates the word cloud to provide an overview of key concepts under discussion in that moment.

Sensors such as eye-tracking sensors, EEG sensors, and inertial measurement units provide data that allow additional dimensions to be considered and new insights gleaned. VizScribe provides a general framework for the visualization and analysis of rich datasets, allowing the researcher to create meaningful visualizations of new data forms (as shown in Figure 9), that are linked to traditional text and video data. We previously discussed how this enables functions such as *filtered navigation*, *multiple approaches to explore datasets*, and *temporal correlation of multiple data streams* that allows participants to access, interpret, and integrate sensor data. VizScribe thus moves beyond existing computer-based qualitative analysis tools to provide support for analyzing both traditional as well as emerging forms of qualitative data.

Qualitative data analysis *requires* that the analyst familiarize themselves with every bit of the data. VizScribe provides a framework to allow a deeper, holistic, and integrated analysis and interpretation of this data and its connection to other related data. It supports micro-level analysis of specific temporal slices, using crosssectional data as well as macro-level longitudinal timeline analysis of aggregated data. By viewing synchronized and integrated data and understanding them in specific temporal and textual contexts, interpretations of design behavior are more fully contextualized and better reflective of the design process.

11. Limitations and Future Work

VizScribe has been designed for navigation and coding of data that otherwise cannot be easily visualized into a single, dashboard view. To an extent, this requires pre-processed data, for example, all data needs to be time-stamped, transcripts benefit from speaker identification, and so on. While the coded data can be exported from VizScribe for additional analyses, VizScribe would be a more effective tool if it offered end-to-end processing and analytics required for qualitative analysis. We envision the following future enhancements to VizScribe:

Preprocessing: This involves both the implementation of algorithms such as the Penn Phonetics Lab Forced Aligner (Yuan & Liberman, 2008) for synchronizing the transcript to the audio track, as well as allowing interactive selection and customization of the information visualizations used. The current implementation allows customization when performed programmatically. However, based on the data imported (keylogs, browsing data, biometric data), the user should be allowed to interactively select appropriate visualizations.

Data-aware annotation and coding: VizScribe currently has the framework for allowing multiple timeline visualizations, but not timeline-level coding. This is particularly important for coding sketches, movements across the design space, gestures, and so on. Our future plans include a data representation of codes applied directly on the timeline, and consolidated together with all representations. Additionally, the absence of data—be it radio silence, video inactivity, or audio silence—is often as significant as the presence of data. Thus, gaps in the visualizations, where no geometric or text entity is displayed, need also be amenable to navigation and coding.

Analytics: The next step for VizScribe would be to incorporate visual analytics to process the data for a higher-level exploration. This includes, but is not limited to, features such as named entity recognition and tagging, parts-of-speech tagging, semantic distance-based filtering of the transcript based on the defined codes. Sociometric data would also benefit from such analytics to better support embodied and socio-material interactions. Additional use of analytics would be in processing the coded data to provide meaningful results. This can include inter-coder reliability calculations and axial coding, which includes the ability to generate matrices to compare instances tagged with intersecting codes.

Collaborative coding: VizScribe's web-based implementation allowed us to provide a platform-independent solution making effective use of data-driven representations that are now available for the web browser. However, this client-server framework also gives us the ability to enable collaborative coding. Participants should be able to log in to access the same dataset and compare their coding process with others. This can find application in training novices in the coding process, where they can overlay their codes with that of an expert for comparison.

Dissemination: With visualization toolkits such as D3, the web browser has already become a medium for both generation and dissemination of visual representations. Future work in VizScribe will also look at means to export the final data and analyses into reports or presentations.

12. Conclusion

We have outlined the requirements of design protocol analysis tools, emphasizing the need for building custom timeline views so that the analyst can set up their own visual representations of design activity. We then presented VizScribe, a visual analytics-based framework for representing and exploring design protocols. Our framework bridges a gap that is currently not addressed by existing qualitative data analysis tools, namely, the processing and presentation of new and emerging forms of data such as sensor data and user log data. VizScribe imports video, transcript, and other log data, and uses linked and interactive representations for the user to navigate and explore, code, and export the data. We refined and enhanced visualizations and interaction modes of the framework by conducting formative studies. We then performed a summative evaluation of VizScribe, through which we showed the advantages of (a) filtered navigation, in helping identify context-specific patterns in the multimodal data visualizations, (b) multiple ways for analysts to approach the same dataset, and (c) supporting the identification of verbal and non-verbal relations within datasets. Finally, we found that the transcript, timeline, and sketch views were versatile visualizations, with custom views bringing newer ways of navigating the dataset and obtaining newer insights.

Acknowledgements

This work is partly supported by the U.S. National Science Foundation under grants IIS-1227639, IIS-1249229, IIS-1253863, IIS-1422341, and IIS-153953, and the Donald W. Feddersen Chaired Professorship at the School of Mechanical Engineering at Purdue. The authors are also thankful to the many participants for their time, patience, and suggestions to improve this work, as well as to the reviewers for their insightful comments and suggestions.

References

André, P., Wilson, M. L., Russell, A., Smith, D. A., Owens, A. et al. (2007). Continuum: designing timelines for hierarchies, relationships and scale. In *Proceedings of the ACM Symposium on User Interface Software and Technology* (pp. 101–110).

Bangor, A., Kortum, P., & Miller, J. (2009). Determining what individual SUS scores mean: Adding an adjective rating scale. *Journal of usability studies*, *4*, 114–123.

Bi, Y., Shergadwala, M., Reid, T., & Panchal, J. H. (2015). Understanding the utilization of information stimuli in design decision making using eye gaze data. In *Proceedings of the ASME International Design Engineering Technical Conferences and Computers and Information in Engineering Conference* (pp. V02AT03A017–V02AT03A017).

Bostock, M., Ogievetsky, V., & Heer, J. (2011). D^3: Data-driven documents. *IEEE Transactions on Visualization and Computer Graphics*, *17*, 2301–2309.

Brooke, J. (1996). SUS-a quick and dirty usability scale. *Usability evaluation in industry*, (pp. 189–194).

Brundell, P., Tennent, P., Greenhalgh, C., Knight, D., Crabtree, A., O'Malley, C., Ainsworth, S., Clarke, D., Carter, R., & Adolphs, S. (2008). Digital replay system (DRS)–a tool for interaction analysis. In *Proceedings of the International Conference on Learning Sciences (Workshop on Interaction Analysis)*.

Carpendale, M. (2003). *Considering visual variables as a basis for information visualization*. Technical Report 2001-693-16, Department of Computer science, University of Calgary Calgary, AB, Canada.

Collins, C., Carpendale, S., & Penn, G. (2009a). DocuBurst: Visualizing document content using language structure. *Computer Graphics Forum*, *28*, 1039–1046.

Collins, C., Viegas, F., & Wattenberg, M. (2009b). Parallel tag clouds to explore and analyze faceted text corpora. In *IEEE Symposium on Visual Analytics Science and Technology* (pp. 91–98).

Corbin, J. M., & Strauss, A. (1990). Grounded theory research: Procedures, canons, and evaluative criteria. *Qualitative sociology*, *13*, 3–21.

Creswell, J. W. (2012). *Qualitative inquiry and research design: Choosing among five approaches*. Sage.

Diederich, J., Ruhmann, I., & May, M. (1987). KRITON: a knowledge-acquisition tool for expert systems. *International Journal of Man-Machine Studies*, *26*, 29–40.

Dinar, M., Shah, J. J., Cagan, J., Leifer, L., Linsey, J., Smith, S. M., & Hernandez, N. V. (2015). Empirical studies of designer thinking: Past, present, and future. *Journal of Mechanical Design*, *137*, 021101.

Dong, A. (2005). The latent semantic approach to studying design team communication. *Design Studies*, *26*, 445–461.

Drisko, J. W. (2013). Qualitative data analysis software: An overview and new possibilities. In A. E. Fortune, W. J. Reid, & R. L. M. Jr. (Eds.), *Qualitative Research in Social Work* (pp. 284–303). (2nd ed.).

Duff, P. A., & Séror, J. (2005). Computers and qualitative data analysis: Paper, pens, and highlighters vs. screen, mouse, and keyboard. *TESOL Quarterly*, *39*, 321–328.

Elmqvist, N., Moere, A. V., Jetter, H.-C., Cernea, D., Reiterer, H., & Jankun-Kelly, T. (2011). Fluid interaction for information visualization. *Information Visualization*, *10*, 327–340.

Ericsson, K. A. (2006). Protocol analysis and expert thought: Concurrent verbalizations of thinking during experts performance on representative tasks. In K. A. Ericsson, N. Charness, P. J. Feltovich, & R. R. Hoffman (Eds.), *The Cambridge Handbook of Expertise and Expert Performance* (pp. 223–241). Cambridge University Press.

Fouse, A., Weibel, N., Hutchins, E., & Hollan, J. D. (2011). Chronoviz: a system for supporting navigation of time-coded data. In *ACM Extended Abstracts on Human Factors in Computing Systems* (pp. 299–304).

Geertz, C. (1973). In *The interpretation of cultures* chapter Thick Description; Toward an lnterpretive Theory of Culture. (pp. 3–30).

Glaser, B. G., & Strauss, A. L. (2009). *The discovery of grounded theory: Strategies for qualitative research.* Transaction Publishers.

Gloor, P. A., Grippa, F., Putzke, J., Lassenius, C., Fuehres, H., Fischbach, K., & Schoder, D. (2012). Measuring social capital in creative teams through sociometric sensors. *International Journal of Organisational Design and Engineering*, *2*, 380–401.

Göker, M. H. (1997). The effects of experience during design problem solving. *Design Studies*, *18*, 405–426.

Goldschmidt, G. (1991). The dialectics of sketching. *Creativity research journal*, *4*, 123–143.

Harrower, M., & Brewer, C. A. (2003). Colorbrewer.org: an online tool for selecting colour schemes for maps. *The Cartographic Journal*, *40*, 27–37.

Heer, J., & Shneiderman, B. (2012). A taxonomy of tools that support the fluent and flexible use of visualizations. *ACM Queue*, *10*, 1–26.

Henderson, K. (1998). The role of material objects in the design process: A comparison of two design cultures and how they contend with automation. *Science, Technology & Human Values*, *23*, 139–174.

Hix, D., & Hartson, H. R. (1993). *Developing User Interfaces: Ensuring Usability Through Product & Process*. New York, NY, USA: John Wiley & Sons, Inc.

Hoey, M., Scott, M., & Thompson, G. (2001). *Patterns of Text*. John Benjamins Publishing Company.

Huber, G., & Garca, C. (1991). Computer assistance for testing hypotheses about qualitative data: The software package AQUAD 3.0. *Qualitative Sociology*, *14*, 325–347.

Isenberg, P., Tang, A., & Carpendale, S. (2008). An exploratory study of visual information analysis. In *Proceedings of the ACM Conference on Human Factors in Computing Systems* (pp. 1217–1226).

Kim, T., Chang, A., Holland, L., & Pentland, A. S. (2008). Meeting mediator: enhancing group collaboration with sociometric feedback. In *The ACM Extended Abstracts on Human Factors in Computing Systems* (pp. 3183–3188).

Kim, T., McFee, E., Olguin, D. O., Waber, B., & Pentland, A. (2012). Sociometric badges: Using sensor technology to capture new forms of collaboration. *Journal of Organizational Behavior*, *33*, 412–427.

Kudrowitz, B. M., & Wallace, D. R. (2010). The play pyramid: A play classification and ideation tool for toy design. *International Journal of Arts and Technology*, *3*, 36–56.

Li, Q., & North, C. (2003). Empirical comparison of dynamic query sliders and brushing histograms. In *Proceedings of the IEEE Symposium on Information Visualization* (pp. 147–153).

Lu, C.-J., & Shulman, S. W. (2008). Rigor and flexibility in computer-based qualitative research: Introducing the coding analysis toolkit. *International Journal of Multiple Research Approaches*, *2*, 105–117.

Manning, C., & Schütze, H. (1999). *Foundations of Statistical Natural Language Processing*. MIT Press.

McPherson, J., Ma, K.-L., Krystosk, P., Bartoletti, T., & Christensen, M. (2004). PortVis: a tool for port-based detection of security events. In *Proceedings of the ACM Workshop on Visualization and Data Mining for Computer Security* (pp. 73–81).

Monroe, M., Lan, R., Morales del Olmo, J., Shneiderman, B., Plaisant, C., & Millstein, J. (2013). The challenges of specifying intervals and absences in temporal queries: a graphical language approach. In *Proceedings of the ACM Conference on Human Factors in Computing Systems* (pp. 2349–2358).

Müller-Tomfelde, C. (2007). Dwell-based pointing in applications of human computer interaction. In C. Baranauskas, P. Palanque, J. Abascal, & S. Barbosa (Eds.), *Human-Computer Interaction* (pp. 560–573). volume 4662 of *Lecture Notes in Computer Science*.

Newell, A. (1966). On the analysis of human problem solving protocols. In *International Symposium in Mathematical Methods in the Social Science*.

Oak, A. (2011). What can talk tell us about design?: Analyzing conversation to understand practice. *Design Studies*, *32*, 211–234.

Olguín, D. O., Waber, B. N., Kim, T., Mohan, A., Ara, K., & Pentland, A. (2009). Sensible organizations: Technology and methodology for automatically measuring organizational behavior. *Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on*, *39*, 43–55.

Olsson, J., & Boldt, M. (2009). Computer forensic timeline visualization tool. *Digital Investigation*, *6, Supplement*, S78–S87.

Plaisant, C., Milash, B., Rose, A., Widoff, S., & Shneiderman, B. (1996). LifeLines: visualizing personal histories. In *Proceedings of the ACM Conference on Human Factors in Computing Systems* (pp. 221–227).

Pourmohamadi, M., & Gero, J. S. (2011). LINKOgrapher: An analysis tool to study design protocols based on FBS coding scheme. In *Proceedings of the International Conference on Engineering Design*.

Robinson, A. C. (2008). Collaborative synthesis of visual analytic results. In *Proceedings of the IEEE Symposium on Visual Analytics Science and Technology* (pp. 67–74).

Rosé, C., Wang, Y.-C., Cui, Y., Arguello, J., Stegmann, K., Weinberger, A., & Fischer, F. (2008). Analyzing collaborative learning processes automatically: Exploiting the advances of computational linguistics in computer-supported collaborative learning. *International Journal of Computer-Supported Collaborative Learning*, *3*, 237–271.

Rubin, S., Berthouzoz, F., Mysore, G. J., Li, W., & Agrawala, M. (2013). Content-based tools for editing audio stories. In *Proceedings of the ACM symposium on User interface software and technology* (pp. 113–122).

Sanderson, P. M., James, J. M., & Seidler, K. S. (1989). SHAPA: an interactive software environment for protocol analysis. *Ergonomics*, *32*, 1271–1302.

Shneiderman, B. (1996). The eyes have it: A task by data type taxonomy for information visualizations. In *Visual Languages, 1996. Proceedings., IEEE Symposium on* (pp. 336–343).

Stab, C., Nazemi, K., & Fellner, D. W. (2010). SemaTime – timeline visualization of time-dependent relations and semantics. In *Advances in Visual Computing* (pp. 514–523). Springer volume 6455 of *Lecture Notes in Computer Science*.

Stasko, J., Görg, C., & Liu, Z. (2008). Jigsaw: supporting investigative analysis through interactive visualization. *Information visualization*, *7*, 118–132.

Thomas, J. J., & Cook, K. A. (2005). *Illuminating the path: The research and development agenda for visual analytics*. Technical Report.

Ullman, D. G., Wood, S., & Craig, D. (1990). The importance of drawing in the mechanical design process. *Computers & graphics*, *14*, 263–274.

Wang, T. D., Plaisant, C., Quinn, A. J., Stanchak, R., Murphy, S., & Shneiderman, B. (2008). Aligning temporal data by sentinel events: discovering patterns in electronic health records. In *Proceedings of the ACM Conference on Human Factors in Computing Systems* (pp. 457–466).

Ware, C. (2012). *Information visualization: perception for design*. Elsevier.

Waterman, D. A., & Newell, A. (1973). PAS-II: an interactive task-free version of an automatic protocol analysis system. In *Proceedings of the ACM Conference on Artificial intelligence* (pp. 431–445).

Wattenberg, M. (2002). Arc diagrams: Visualizing structure in strings. In *IEEE Symposium on Information Visualization* (pp. 110–116).

Wattenberg, M., & Viégas, F. B. (2008). The Word Tree, an interactive visual concordance. *IEEE Transactions on Visualization and Computer Graphics*, *14*, 1221–1228.

Yi, J. S., ah Kang, Y., Stasko, J. T., & Jacko, J. A. (2007). Toward a deeper understanding of the role of interaction in information visualization. *IEEE Transactions on Visualization and Computer Graphics*, *13*, 1224–1231.

Yuan, J., & Liberman, M. (2008). Speaker identification on the SCOTUS corpus. *Journal of the Acoustical Society of America*, *123*, 3878.

Zhao, Z., Badam, S. K., Chandrasegaran, S., Park, D. G., Elmqvist, N., Kisselburgh, L., & Ramani, K. (2014). skWiki: a multimedia sketching system for collaborative creativity. In *Proceedings of the ACM Conference on Human Factors in Computing Systems* (pp. 1235–1244).