Chapter in Software Designers in Action: A Human-Centric Look at Design Work, Marian Petre, Andre Van Der Hoek (Eds.), Chapman & Hall, September, 2013.

Conjectures on How Designers Interact with Representations in the Early Stages of Software Design

Kumiyo Nakakoji (Software Research Associates, Inc.)
Yasuhiro Yamamoto (Tokyo Institute of Technology)

1. Introduction

The three design sessions being studied in the Studying Professional Software Design (SPSD) Workshop [Petre et al. 2010] demonstrate a wide variety of activities, with participants given the same design prompt with the same physical setting in which two designers work on the task in front of the whiteboard. The differences are not in the approaches that were taken, but in the aspects of software design the participants worked on. The variety of the three design processes seems to reflect the wide range of activities software design involves and the different levels of detail that need to be covered in software design.

Studies have been published about the three design sessions, including those that focus on how design process proceeds through a fine-grained analysis [Baker et al. 2010], how a pair of designers collaborate with each other through an ethnographic analyses [Rooksby, Ikeya 2012], and how designers make decisions through a design protocol analysis [Christiaans, Almendra, 2010].

This chapter focuses on how designers interact with representations in the early stages of software design. Each team worked on the design problem by talking to each other and using the whiteboard as a means of externalization [Yamamoto & Nakakoji 2005]. The drawings on the whiteboard captured in the video data and the transcript of the conversations provide a rich source of representations with which the designers interacted.

The focus of the study presented here is similar to that of Schoen [Schoen 1983], who observed how two architectural designers engaged in sketching and talking about the design problem. We are interested in exploring the dynamism of the representations externalized by the designers in a practical setting, viewed as the designers' conversations with the material [Schoen 1983]. Our work is distinctive from those that analyze the sketched diagrams (e.g., [Do 2005]), or those that analyze cognitive aspects of designers in controlled study settings (e.g., [Suwa, Tversky 2001]).

Our study primarily focuses on the whiteboard usage together with word usage in terms

of which were generated and then used: what diagrams were drawn on the whiteboard, what text was put on the whiteboard, and what phrases were uttered in the transcript, as well as how these communications were referred to, revisited, modified, and re-appropriated in the subsequent process. Ju and colleagues [Ju et al. 2006] reported on how people use a whiteboard in a collaborative setting and Walny and colleagues [Walny et al. 2011] analyzed what they called spontaneous visualizations on the whiteboard.
This chapter analyzes the use of a whiteboard along with textual representations uttered and written down on the whiteboard during the course of software design practice.

2. Data Analyses Setting

In analyzing data that consist of meeting videos and their transcripts, we use Design Practice Streams (DPS) tools [Nakakoji et al. 2012]. DPS has three components: MovieViewer, StrokeViewer, and TranscriptViewer (Figure 1). DPS helps us to browse segments of the video data relevant to the focused topic by specifying a region on the whiteboard or choosing a few terms used in the transcript.

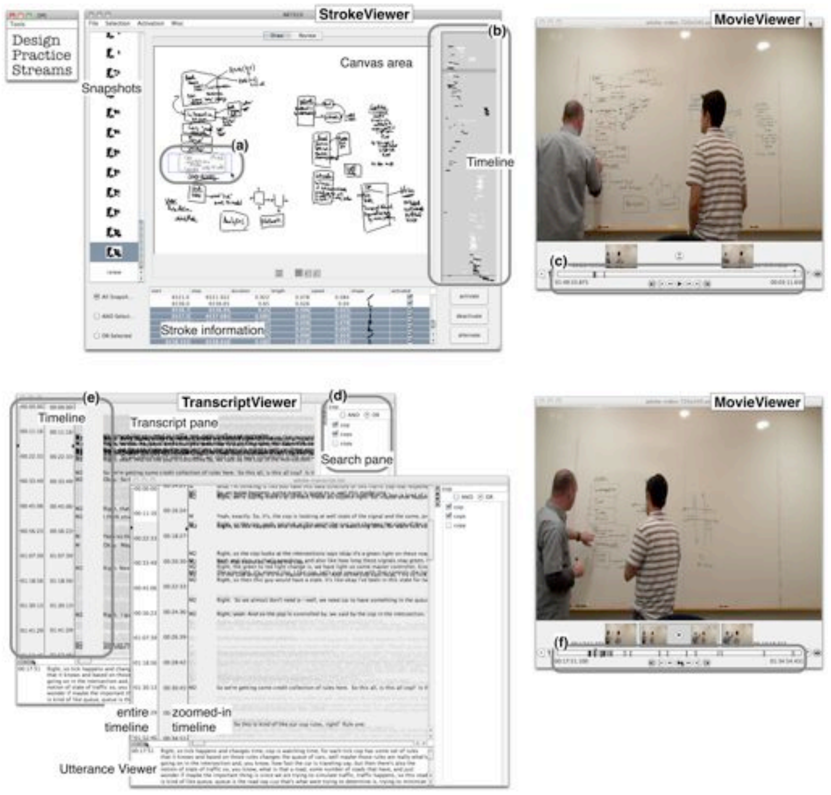


Figure 1: DPS (Design Practice Streams) tools showing the Adobe data

DPS uses a simple timestamp mechanism to relate the video data of a design meeting with the digital whiteboard drawing data (i.e., a set of timestamped strokes drawn during the design meeting) and the textual transcribed data (i.e., a set of timestamped utterances). DPS does not require any manual tagging, annotations, or semantic interpretations. The detailed mechanism of DPS is described in [Nakakoji et al. 2012].

The original dataset did not include the whiteboard stroke data, so we manually generated a set of stroke data for the videotaped whiteboard of one of the three teams (Adobe) in sync with the video data. By using DPS to browse the Adobe team design process, we found that the diagrams drawn on the whiteboard evolved over time – not necessarily in a consecutive manner but intermittently, for example, by adding labels to verbally refer to some parts of the diagram, or by adding more graphic representations to reappropriate the diagram for a purpose that is different from the original one [Nakakoji et al. 2012]. We used DPS for the video and the transcript data to analyze the design processes of the other two teams.

In what follows, we list a set of observations that we made in analyzing the data. We first identified distinctive elements of how the designers interact with the representations by viewing the three sets of design session data with DPS and recording the timestamp of each element. We then textually annotated each of such elements regarding its peculiar aspect. Finally, we categorized the elements into groups by spatially arranging them while applying a variation of the KJ (Kawakita Jiro) method [Kawakita 1968] (Figure 2).

Figure 2: Data analysis setting

The results consist of the following topics:

- Observation-A: using the whiteboard as a medium for design
- Observation-B: working with an emerging concept that didn't exist in a given specification
- Observation-C: using the whiteboard for temporal drawing
- Observation-D: producing a variety of notations on the whiteboard
- Observation-E: reappropriating the representations on the whiteboard
- Observation-F: detailing user interface representations
- Observation-G: recognizing an "eureka" moment
- Observation-H: experiencing a temporal gap between talking and drawing
- Observation-I: observing courses of design

- Observation-J: redrawing on the whiteboard

We do not claim here that the observations are exhaustively listed features of the software design. Rather, we argue that the list gives us a partial pre-understanding about how designers interact with externalized representation in the early stages of software design. Some may be cases peculiar to the three design teams and the given environmental settings, some may be specific to the domain of software design, and some may be applicable to other design practices.

3. Observations

The three teams we observed are the Adobe, AmberPoint, and Intuit teams. The following scenarios provide examples of each type of observation.

Observation-A: Using the whiteboard as a medium for design

Designers used a whiteboard drawing as a snapshot to hold their intermediate design ideas.
Jim of the AmberPoint team wanted to take a picture of the whiteboard (AmberPoint: 00:48:35) by saying, “But I don’t want to [erase it], that’s always one of my concerns of writing on a whiteboard is that I’ve got some great ideas on here but now I need to reshuffle a little bit, and so I would just pull out a camera” (00:48:47). He actually did not erase anything right away. The purpose of taking a picture of the whiteboard seemed to be to record a snapshot of the important artifact that he had been working on so that he could reshuffle the ideas in the subsequent process.

Designers looked back at what they had written on the whiteboard.
The designers articulated the process they had used. Male 1 of the Intuit team summarized what they had just done and commented on the process by saying, “I like jumping into details and then coming up and saying, ‘Okay, now what do we really need here, what are we seeing here?’” (Intuit: 00:14:51).

Designers used “designing” and “drawing” as synonyms.
Facing the whiteboard, Ania of the AmberPoint team asked a question: “What else are we going to be drawing?” (AmberPoint: 00:09:33). Jim, also facing the whiteboard, posed a question: “Are there other objects?” (00:10:52). They seem to be saying the same thing but stating it differently – one in terms of things to draw in a graphical user interface (GUI) window, and the other in

terms of objects to list.

Designers walked through user interaction steps by interacting with a drawn GUI on the whiteboard.

Designers simulated a short interaction flow with the drawn GUI windows on the whiteboard by using gestures. When Jim of the AmberPoint team was saying, "So the user can change where the intersections are, the distance between them, by moving them around," he moved both of his hands and showed a gesture expanding the road drawn in the user interface window on the whiteboard.

While sketching a GUI, M of the Adobe team kept talking about how a user would interact with the system, such as, "So once you drag this one here you have to specify how that one's connected, right? So as you drag, as you drag each intersection on it will snap and say, okay, I'm connected to here and to here, and it snaps. And here, and what not. And so then drag it and then it automatically connects and as you move this back and forth it increases the capacity for this road so now it …" (Adobe: 01:05:45). While doing so, M made gestures pointing to and touching the GUI window drawn on the whiteboard.

The designers occasionally acted as if there had been something drawn on the whiteboard.

When discussing how a student would interact with the system, Ania of the AmberPoint team used gestures interacting with the whiteboard, and relayed possible interaction steps by saying, "So there is some sort of drawing palette, right, that says okay, I have this thing I drag something, I'm drawing a road and I call it something and I draw and I call it B ..." (AmberPoint: 00:20:10). Ania used the object listed on the left side of the whiteboard as the palette, although it was just the list of objects and was not drawn as the palette.

Observation-B: Working with an emerging concept that didn't exist in a given specification

A new concept that is not a part of the given specification became a major design element.

When trying to understand how cars go through an intersection, M of the Adobe team mentioned "cop" for the first time (Adobe: 00:14:06) by saying, "It's almost like you need [a] traffic cop, right, traffic cop controller." The term "cop" does not appear in the given task prompt.

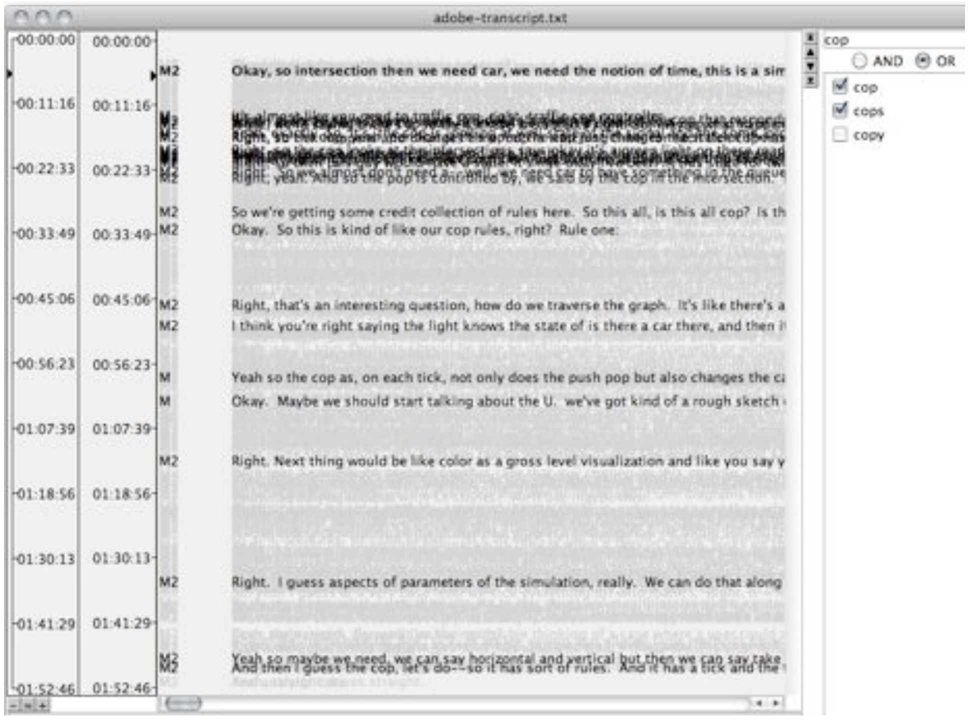
After M mentioned the term, M2 drew a larger square around the model and its components on the whiteboard (00:14:16). He added a box underneath the square he had just drawn as if he was ready to add an "external" entry to the "model." M2 wrote "Event (e)" in the new box, saying, "So, something happens as an external controller is ticking the model, is that what you're saying?" (00:15:07). Then, M2 wrote "cop" on the whiteboard near the "model" label (00:15:15), and then M said, "Well, I don't know, is the cop really a model or is that a controller? A cop is kind of controlling the state of the model" (00:15:18). M2 then immediately erased it (00:15:26). He then deleted the square frame surrounding the model field.

After pondering for a few seconds, M2 started boxing each of "intersection," "car," and "time" by saying, "These are objects, right?" (00:15:29). He deleted "Event" that he kept unboxed.

When M2 wrote "cop" again on the whiteboard (00:16:37), he wrote "responds to time changes" and "changing state of model" by verbalizing them. He then wrote a square around the "cop" and its description, and asked M if M thought it was reasonable.

This resulted in the list of boxed terms under the model field, and "cops" had become a boxed object under the "model" category. Thus, "cop" had become an object in their model (see Figure 3(a)).

The "cop" object served as an important player in their model in the subsequent design process. The terms "cop" and "cops" were mentioned in 28 utterances in the transcript, ranging over the entire design process (see Figure 3(b)).



```
adobe-transcript.txt                                                                 cop
                                                                                      ○ AND  ◉ OR
00:00:00  00:00:00                                                                    ☑ cop
                   M2   Okay, so intersection then we need car, we need the notion of time, this is a sir   ☑ cops
00:11:16  00:11:16                                                                    ☐ copy
                   M2   Right, so the cop, and the change
                   M2   Right. So we almost don't need a-- well, we need car to have something in the queue
00:22:33  00:22:33 M2   Right, yeah. And so the pop is controlled by, we said by the cop in the intersection.
                   M2   So we're getting some credit collection of rules here. So this all, is this all cop? Is th
00:33:49  00:33:49 M2   Okay. So this is kind of like our cop rules, right? Rule one:

00:45:06  00:45:06 M2   Right, that's an interesting question, how do we traverse the graph. It's like there's a
                   M2   I think you're right saying the light knows the state of is there a car there, and then i

00:56:23  00:56:23 M    Yeah so the cop as, on each tick, not only does the push pop but also changes the ca
                   M    Okay. Maybe we should start talking about the U. we've got kind of a rough sketch
01:07:39  01:07:39
                   M2   Right. Next thing would be like color as a gross level visualization and like you say y
01:18:56  01:18:56

01:30:13  01:30:13
                   M2   Right. I guess aspects of parameters of the simulation, really. We can do that along
01:41:29  01:41:29

                   M2   Yeah so maybe we need, we can say horizontal and vertical but then we can say take
                   M2   And then I guess the cop, let's do--so it has sort of rules. And it has a tick and the
01:52:46  01:52:46
```

Figure 3: “Cop” drawn on the whiteboard (a) and uttered in the transcript (b)

A new concept was introduced to help the designers understand the problem better, but it did not become a design element.

When trying to describe how cars come and go through an intersection in the traffic simulation, the Intuit team had come up with the term “faucet.” When wondering what would be the “start” and “stop” of the simulation, Male 1 of the Intuit team used “faucet” as a metaphor to describe the behavior of the controller, as in the following: "I'd imagine some guy turning on a faucet. When the sink's full, turn it off, let it drain down, or tell me there's more room, I'll turn it on again" (Intuit: 00:41:35). This metaphor seemed to be a comprehensive one, and Male 2 agreed with Male 1 by saying, "Perfect, okay, yeah, that makes sense" (00:41:58). In contrast to the "cop" object with the Adobe team, the Intuit team did not make "faucet" a part of their model.

Designers spent a long time deciding to make a new concept as a design element.
When Male 1 of the Intuit team drew a line between "Roads" and "Lane" by saying, "Roads have lanes" (Intuit: 00:11:01), Male 2 asked whether they cared about such details as lanes. Then Male 2 concluded by saying, "I don't think we need to make assumption of lanes, period" (00:12:11).

Despite this decision, the notion of "Lane" kept emerging in the subsequent discussions. When they finished discussing the length of the road in front of the pictures of the two intersections on the whiteboard, Male 1 walked toward the left of the whiteboard, where "Roads" was listed as "Data." He then annotated "Lane" connected to "Roads" with a cross in red, by saying, "So we've decided that lanes may be not so significant but length is going to be significant" (00:19:38). He continued by saying, "And it's significant because it holds a number of cars" (00:19:40), as he drew a red line starting from "Roads" and wrote "Length" and "# of cars."

Male 1 then said, "I'm still wondering about the left-turn lane" (00:20:21). After describing why having lanes affect the complexity of the problem, he claimed to have "a protected left" to "go with the minimum" (00:21:03) in taking the simplest assumption. The two continued discussing how having lanes affects the traffic simulation for a few more minutes. Male 1 then expressed his concern by saying, "I hate to bog down in that detail."

Finally, in about 30 minutes, when Male 1 went back to where he wrote "Roads" under "Data," he added "s" to "Lane" connected to "Roads" to make "Lanes" (00:48:59). He then wrote "L/S/R" near the "Lanes," with Male 2 saying, "More paths, kind of" (00:49:08). This way they seemed to have agreed that lanes are like "paths" for each of the directions (left, straight, right).

Observation-C: Using the whiteboard for temporal drawing

Designers casually drew simple diagrams on the whiteboard to quickly establish shared understanding.

M of the Adobe team drew a graph diagram when explaining the relationship between "capacity" and "wait time" (01:33:30). He erased the graph immediately after he finished his explanation (Adobe: 01:34:02).

When M2 of the Intuit team asked whether "all the opposing lanes [were] always the same state" (Intuit: 01:48:25), M drew a picture of a simplified intersection in a corner space, trying to illustrate the real situation, and described that "it [what M2 said] is a simplified assumption" (01:48:29). M then immediately erased the intersection that he had just drawn.

Designers used diagrams to understand a real-world phenomenon.

M2 of the Adobe team drew a picture of an intersection on the whiteboard after a few minutes of talking about the model (Adobe: 00:06:41). This picture was used not so much as an image of the map visualization but as an object-to-think-with in understanding the role and behavior of an intersection and its relationships with cars, lanes, and signals.

The Intuit team drew diagrams as a representation of the world to talk about a model for the world. Male 1 drew a picture of an intersection (i.e., crossing roads) and added visual marks corresponding to the components of "Data" that he had just written, such as cars and signals, while talking about how they were related to one another (Intuit: 00:01:00). Then they added arrows and little marks on the intersection diagram they had drawn when talking about Data. While discussing, they wrote "Abstract," underlined it, wrote "rules" beneath "Abstract" (Intuit: 00:13:00), and drew a line between "rules" and "signals." As they talked about the objects listed under "Data" and "Abstracts" and their relationships by adding lines, they tried to understand the world of the traffic system; as Male 1 said, "So we [are] kind of tinkering around with this real-world model" (00:14:41).

Observation-D: Producing a variety of notations on the whiteboard

Designers used specific notations to represent the type or function of a diagram drawn on a whiteboard.

The designers used boxes, lines, and arrows as specific notations to represent objects and hierarchical relationships. M2 of the Adobe team wrote down “clock controller” under the label “controllers,” and boxed the words. Then M2 drew an arrow and wrote “send ‘tick’ event to model” (Adobe: 00:13:40). This seems to become the description of what a “clock controller” would do.

In talking about the “hierarchy,” Male 1 of the Intuit team draw a line between “interaction” and “signal,” indicating that there is a hierarchical relationship between the two.

Designers used a pseudo-code expression to describe the system to be designed.

In summarizing what they had designed, M2 of the Adobe team wrote in a programming-language-like notation, “main( ) { n = new network, c = new clock. c.run(n) }” and described it as “at a very gross level” (Adobe: 01:37:50).

When M2 of the Adobe team wrote down each rule for the “cop rules,” he wrote it quite in detail by using a pseudo programming-language-like notation (Adobe: 00:34:32), such as, “1 car per tick passes if car is the head of the road” when “car == straight and I.light == greenlight R.pop” (00:36:15). When he wrote “I.light=green” on the whiteboard, he actually was saying “And green, and intersection I.light.” He then put “(1)” in the head of the line he just wrote, saying “So that’s case one, that’s case one. Case two would be... “ (00:36:06) and put “(2)” in the new line.

Designers used colors on the whiteboard to match the GUI colors and to distinguish different concepts or phases.

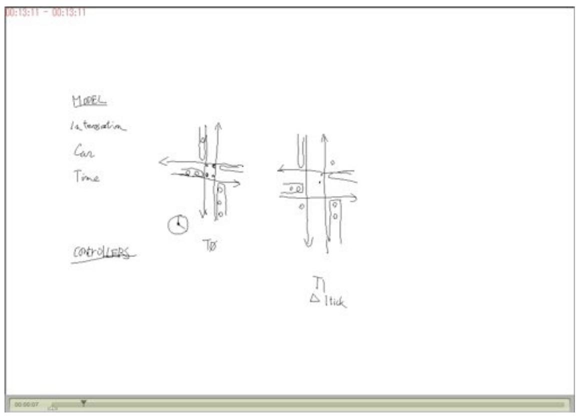
While drawing the timeline, Jim of the AmberPoint team used a green pen for a green light, a red pen for a red light, and a yellow pen for a yellow light. He asked for another color by saying, “That represents green arrow if we need to. Left-turn is orange, I don’t know.” He then used a purple pen (AmberPoint: 00:39:58).

When they started thinking about “container,” Male 1 of the Intuit team wrote “map” and underlined it with a green pen in the space in the top part of the whiteboard (Intuit: 00:30:50). He then circled “interactions” and “Roads” connected from “map” (00:31:07) also in green. Male 1 kept using a green pen and drew two circles and a connecting line, and labeled them “I1,” “I2,” and “P1,” respectively (00:33:50). He then asked, “How do you define the behavior of the traffic?” (00:34:30). He then wrote “behavior” in the bottom part of the whiteboard in green as if it would serve as a reminder to rethink later the things written in green (00:34:50).

Observation-E: Reappropriating the representations on the whiteboard

Designers re-appropriated a drawn diagram on the whiteboard.

A diagram of two intersections was first drawn when the two designers of the Adobe team were discussing how the state of an intersection changes from time T0 to time T1 (Adobe: 00:09:57; see Figure 4(a)). About 30 minutes later, when the designers were talking about how cars flow from one intersection to another, they started pointing to the drawn intersections as if they were adjacent to each other. M2 then drew dotted lines to connect the two intersections (00:40:33; see Figure 4(b)). The drawing resulted in a discrepancy between labels (T0 and T1) and dotted lines.



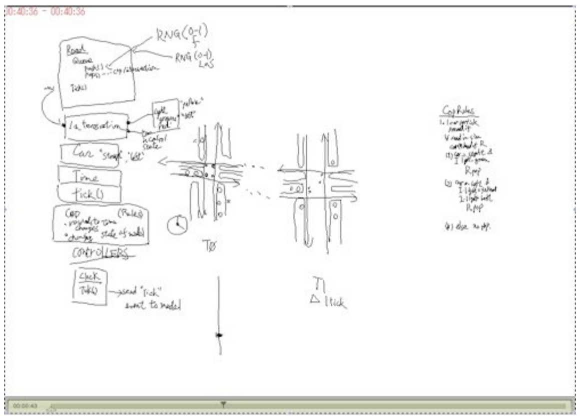


Figure 4: Intersections reappropriated – initial drawing (a) and later drawing (b)

Designers reappropriated written text on the whiteboard.

When designers were re-examining the "rules" of signals, the two designers of the Intuit team discussed the relation between "time" and "red/yellow/green," both of which had been written under "rules" (Intuit: 00:51:32). Male 2 said, "There should be a time component attached to each one of these things," pointing to "red/yellow/green." Then Male 1 added a "t-" to each of "red/yellow/green" to create "t-red/t-yellow/t-green," and said, "There's a time for each of those. t-red, t-yellow, t-green" (00:52:09). Thus, "red/yellow/green," which was originally written to mean something to do with rules related to the three options, had come to mean the three time durations for the red, yellow, and green lights when each was prefixed with "t-."

Observation-F: Detailing user interface representations

Designers went into extreme detail in sketching GUI components on the whiteboard.
Jim of the AmberPoint team started drawing how the "summary area" would look by writing down what information they would like to see in the area (AmberPoint 00:14:00). He listed all the road names and intersections that were drawn on the left-side of the map (00:14:50).

The designers spent a few minutes just redrawing the details. In response to the discussion on how a user would specify the duration of the cycle a signal, Jim of the AmberPoint team erased all the inside area of the pop-up window where he had spent several minutes listing the conditions, and started to redraw them on the whiteboard (AmberPoint: 00:36:30).

Designers detailed not only in drawing but also in talking.
In sketching the pop-up window for an intersection to specify the signal behavior, the AmberPoint team discussed details such as "2 minutes" for the default cycle duration of a signal (AmberPoint: 00:29:59), and "on average 40 seconds because with the green arrow" (00:30:42), and "so it would be 50 seconds at the top of the cycle followed by, I don't know, 10 seconds of yellow, followed by how about 55 here" (00:31:00). Jim used a green pen to represent a green light, a red pen for a red light, and so on (see Figure 5).

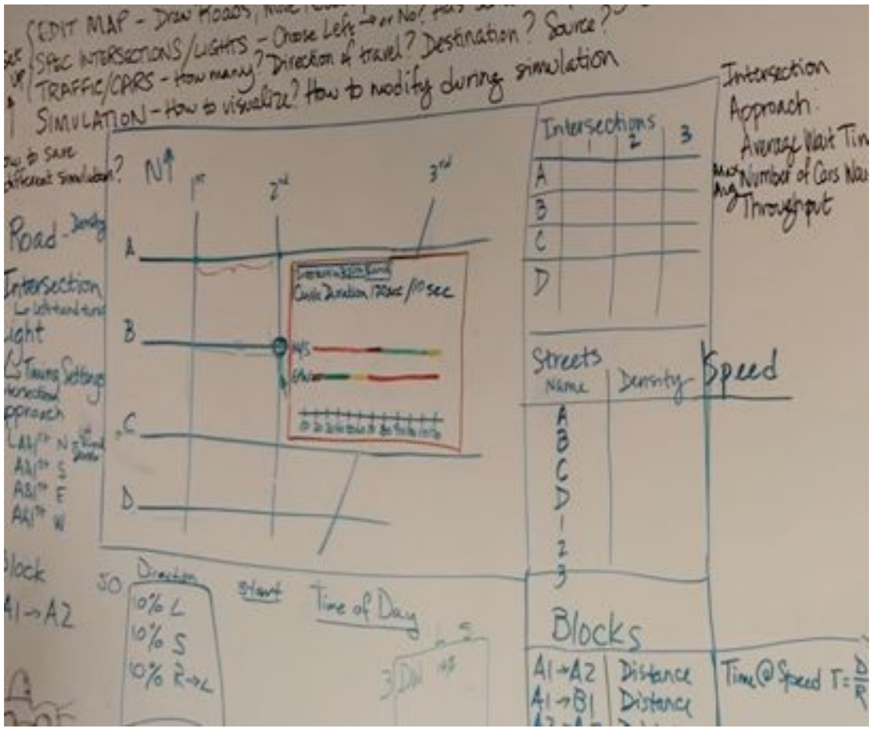


Figure 5: Detailed drawing on the whiteboard

Designers prioritized the completeness of a textual list to the exact layout of GUI components in drawing a GUI.

When Jim of the AmberPoint team added the "speed" column in the drawing of the "summary area" window, the section did not fit within the drawn square area and stuck out from the area toward right, but he left it there (AmberPoint: 00:14:50).

Observation-G: Recognizing an "eureka" moment

Designers came up with a new understanding of the problem through talking.

After talking about making "Roads" own traffic, Male 1 of the Intuit team said, "so the road is an object" (Intuit: 00:36:08) and wrote as operations "add cards" in green next to "Roads" (00:36:25). Male 2 initially nodded for a few seconds, expressing his agreement. But then Male 2 murmured, "Would it add a car?" (00:36:28). Immediately after that, Male 1 echoed Male 2 by saying, "Yeah, I'm trying to get how do we [add a car]?" (00:36:29). This finding led them to talk about "input" to a road in the subsequent discussions.

The AmberPoint team became aware of a problematic situation by walking through a user's interaction with the currently drawn interface. While Jim was talking about a student's walking through interacting with the current user interface and how the information in the summary area would change (AmberPoint: 00:25:35), he had become aware of a possible problem with the summary window by saying, "So it's funny because some of these things are editable and some of them just kind of give you details about how your edits made a change," and he said, "So I think it's a little confusing" (00:26:25).

Jim was redrawing a map window and a window for setting up the intersection, and was about to draw a window for a car/traffic configuration while saying that these were "editing and setting modes." Ania then wanted to have "the map drawing as one thing" but the intersection specification and the car specification "as a part of the simulation" (01:18:49). The two modes then were renamed to "building map" and "building simulation" (01:21:35).

Designers refined a partial design solution based on a new understanding that emerged through the detailed drawing.

Jim continued listing all possible patterns of signals in the pop-up window by first finishing the N/S options and then the E/W options (AmberPoint: 00:34:06). While listing, he had become aware that the signal has one state when both directions are red (00:34:30).

Observation-H: Experiencing a temporal gap between talking and drawing

Designers did not immediately record seemingly important terms on a whiteboard.
The Adobe team first mentioned the notion of "interior" road (Adobe:00:40:25). It was some time later that they added the "interior" and "onramp" properties to the "Road" on the whiteboard (Adobe: 00:42:55).

Designers did not record on the whiteboard a number of GUI ideas verbally discussed.
When the Adobe team discussed the user interface for the system, a number of ideas such as the following were verbally described, but they were never written on the whiteboard. "Right, so this is now a UI piece, and the dot brings up your editor with frequency of, your frequency value. Maybe if you select a car, I mean a road, it allows you to edit. I wonder if we make the capacity of the geometry and you could sort of simulate a highway or something..." (01:11:52).

Designers wrote down text on the whiteboard for possible later use.

The designers wrote down a not-so-certain object on the whiteboard so that they could come back to it later. When the designers of the Intuit team went back to working on the "rules" of signals under "Abstract," Male 2 decided to write "left/straight/right" (00:51:25) under "time," "sensor," and "red/yellow/green" after wondering a little bit. Male 1 agreed with what Male 2 had just done by saying, "Yeah let's put it in there and we'll capture that" (Intuit: 00:51:17).

Observation-I: Observing courses of design

Designers started the design process by listing names as objects on a whiteboard.

Male 1 of the Intuit team first wrote "Data" on the whiteboard, and underlined it. He then added "signals," "Roads," and "cars (traffic)" (Intuit: 00:08:59).

M and M2 of the Adobe team started by working on "basic data structures" (Adobe: 00:05:56). During the conversation, terms that seem to play important roles in the model were mentioned, and M2 started writing them down on the left side of the whiteboard, for example, "intersection," "car," and "time." Other terms that also seem important, such as "queues" (00:08:28), were mentioned but were not written on the whiteboard then.

The AmberPoint team listed object names on the thin left area of the whiteboard. The remaining part consisted of the visual map area and the summary window area, which listed intersections, streets, and blocks, each with a complete list of roads displayed in the map area (AmberPoint: 00:17:33).

Discussions on user interface and those of a model were intertwined.

When talking about how "push" and "pull" were done by "cop," M of the Adobe team wondered, "Well, it does say that something that the user could control" (Adobe: 00:25:32), and the designers went back to the problem description sheet to better understand how the simulation is controlled by a user (e.g., a student) (00:25:47).

When Male 2 of the Intuit team was talking about how input traffic would flow cars into the road, he wrote "input traffic" and drew a circle on the right side of the intersection picture Male 1 previously had drawn. Then Male 2 drew an arrow from the circle into a road, which was a part of the intersection (00:36:40). Male 1 then wrote "R1" as a label to the road, and he used "R1" in saying, "Let's say there's, I'm thinking, the box they're using they click, they're going to define a flow into R1 and then they're going to click 'go'" (00:36:55).

When talking about "queuing and de-queuing of cars" for visualizing cars in the user interface (Adobe: 01:25:34), the Adobe team started talking about the notion of "queue" and went back to talk about the model, such as, "Would that be a property of the queue or a property of the car?" (01:26:44). The subsequent conversation for the next several minutes kept going back and forth between two topics, the user interface visualization and the model.

Observation-J: Redrawing on the whiteboard

Designers added labels to drawn objects on the whiteboard to talk about them.

When the AmberPoint team talked about signals and left-hand turns, Jim labeled each horizontal road with "A," "B," "C," and "D," and each vertical road with "1st," "2nd," and "3rd" (AmberPoint: 00:12:06). Jim started naming an intersection by using these labels, for instance, "At the corner of A and 1st, there's a north approach" (00:12:31).

Designers added a little context to a diagram later in the process.

Jim of the AmberPoint team added the North-up sign in the top-left corner of the sketched map GUI window drawn on the whiteboard (AmberPoint:00:28:17). The designers had been working on the map from the very beginning of the meeting for about 20 minutes then (00:08:21).

Labeling took place not only during the meeting but also after the meeting.

During a break when M2 of the Adobe team stepped away from the room, M looked at the whiteboard and added "visitor" (Adobe: 01:21:03) as a label to the two lines of text, "rule/actions" and "objects/rules," which had been written down about 20 minutes before the break. The term "visitor" was mentioned at the time M2 put the two lines of text on the whiteboard but had not been written down. M seemed to have remembered the context and recorded the context to the two lines.

Designers redrew diagrams on the whiteboard to afford more space.

When drawn objects on the whiteboard evolved, the designers redrew the same diagram in a different space to afford more space. They did not seem to be aware of how the diagrams would evolve in the beginning of drawing them.

M2 of the Adobe team erased the "model" label and wrote "Road" on the whiteboard where "model" had been (Adobe: 00:18:30). By adding properties on to the "Road," M2 deleted "Road" and wrote it again in an upper position on the whiteboard so that it had more space below to write more properties (Adobe: 00:24:05).

M of the Adobe team copied a picture of the clock toward the right to have more space (Adobe: 01:22:03).

Jim of the AmberPoint team enlarged the pop-up window for the intersection toward the right since the area got quite cramped with its contents. (AmberPoint: 00:33:40).

Designers redrew a great many details in copying a GUI window on the whiteboard before erasing them to afford more space.

In response to the discussion on how a user would specify the duration of the cycle of a signal, Jim erased all the inside area of the pop-up window where he had spent several minutes listing the conditions, and started rewriting them (AmberPoint: 00:36:30).

Sectioning of the whiteboard evolved over time.

The designers also "moved" drawn objects on the whiteboard by copying and erasing them when a grouping of objects emerged. Different types of groups were identified in different sections of the whiteboard.

When M mentioned the "controller" (Adobe: 00:09:37), M2 went back to the place on the whiteboard where he had written a list of terms, and put "model" on top of the list and underlined it (00:10:05). Immediately after that, he wrote "controller" in the space below and underlined it. Now the whiteboard had two areas, one labeled "model" and the other labeled "controllers." The initial list of terms became a part of the "model."

A topic change triggered the designers to erase whiteboard drawings.

When the Adobe team started talking about sketching user interfaces, they first tried to find things that they would erase on the whiteboard, but then decided they would not, and started sketching a GUI window in the small space at the bottom of the whiteboard (Adobe: 01:04:46). When they later changed the topic to discuss a user story, they erased the pictures of the two intersections on the whiteboard to create space (01:19:09).

Designers sought more appropriate wording and expressions when writing on the whiteboard. When Jim of the AmberPoint team started listing "objects that they need to deal with" (AmberPoint: 00:09:20) on the left side of the whiteboard, he asked Ania, "Are we going to call them streets or roads?" Ania responded by saying, "I guess roads, intersections" (00:09:31). Jim then wrote down "Roads" on the whiteboard.

When the notion of "leg" emerged (AmberPoint: 00:15:46), Ania first asked, "What would be an easy way to name each leg?" (00:15:51). Jim said, "If this is A1 and this is A2, the leg would be A1 to A2" (00:16:01) and Jim drew "A1→A2" on the left side of the whiteboard under the list of objects (00:16:20).

Jim added "legs" above "A1→A2" on the left side of the whiteboard, and added "legs" to the interface summary area on the right side of the whiteboard. But then he did not seem to like the term "legs" and changed it to "blocks" on the left side of the whiteboard (AmberPoint: 00:16:30).

While working on a user story, M of the Adobe team kept writing natural language sentences by verbalizing – for instance, "I want to evaluate success of each simulation" (Adobe: 01:31:56). M then pondered what expression he should use by saying, "...average capacity on the road or something? Is it capacity or average umm or umm.. capacity means like..." (01:32:04). M2 then said, "Well, capacity is a good one to have because then you can evaluate wait time per, for a given capacity" (01:32:31). M kept the expression "average capacity" on the whiteboard. A few minutes later when M wrote "I want to control the aspects of the intersection" on the whiteboard (01:35:20), M2 rephrased it by saying "I guess 'aspects of parameters of the simulation,' really" (01:35:28). M changed the writing into "I want to control the parameters of simulation" (01:35:50).

Designers kept consistencies among the drawings on the whiteboard when refining labels and layouts.
When Jim of the AmberPoint team decided to use "blocks" instead of "legs," he also changed "legs" to "blocks" on the right side beneath the summary area, drew a window-like square around the "blocks" that he just wrote, and listed possible block names, such as A1→A2, A1→B2, and so on (00:16:40).

Designers did not keep consistencies among the drawings on the whiteboard when refining labels and layouts.

When discussing how an intersection behaves with roads, Male 1 of the Intuit team labeled the four roads of the intersection he initially drew with "R1," "R2," "R3," and "R4." He then used the labels to describe the behavior of the intersection, such as in "The intersection is going to say, okay, my light is green, R1, I mean it's green for 30 seconds – R1, give me 10 cars, we can calculate that, and put them in R3" (Intuit: 00:46:56). He had already written "R1" on a different road of the adjacent intersection at 00:36:55, so there now existed two R1's in the single drawing.

4. Discussions

This section discusses the observational findings listed in the previous section. We believe that the observations described above can help us formulate hypotheses about how the software designers interact with representations, as well as inform the design of tools for software design, such as whiteboarding tools or diagraming editors for software designers.

4.1 Whiteboard usage in software design

Just as [Dekel, Herbsleb 2007] observed with object-oriented design teams, the three design teams demonstrated the use of the whiteboard as a medium to represent the artifacts that they had worked on (see Observation-A). As Ania of the AmberPoint team pointed out in the video that summarized their task, Jim was said to have strong expertise in using a whiteboard. He often seemed to have planned which part of the whiteboard to use for what purposes.

As Petre [2009] argued, designers "used juxtaposition and annotation deliberately and expressively" by sketching on the whiteboard, using both text and diagrams, as well as formal representations (see Observation-D). The notations used on the whiteboard partially adhered to unified modeling language (UML) class diagram notations, but were not limited to the UML, and incorporated many informal representations [Dekel, Herbsleb 2007]. The designers used pseudo-code expressions to specify rules, conditions, and timing on the whiteboard rather than using natural language textual representations. What the designer uttered was transformed into a programming-language-like expression when being written down on the whiteboard.

The designers deliberately chose colors when sketching a GUI window on the whiteboard. The choice sometimes depended on the choice of colors of GUI components, but other times was merely to distinguish different objects and concepts (Observation-D).

4.2 How objects evolve over time

Two of the three observed teams started the course of design by listing nouns on the whiteboard as object names (see Observation-I), as observed by [Dekel, Herbsleb 2007] . The designers started the design process by first identifying names of things that would serve as components for the software to be built (see Observation-B). Such components are likely to become class and object names in software programs. The teams tried not to erase them throughout the design process.

Some objects had straightforwardly been identified (e.g., “Signal” “Road” “Car”), and others had been debated for quite a while (e.g., “Lane” in the Intuit team and “Cop” in the Adobe team). The former seems to be objects and class names that are commonly used by the three teams; the latter seems to be those that are used by a specific team.

The Intuit team spent a long time deciding whether they should list “Lane” as an object, and they did not write it down on the whiteboard (see Observation-B). Their concern was around a conflicting need between introducing a seemingly necessary concept and keeping the model simple. The Intuit team initially thought that they might need “Lanes” as an object but then worried that the notion is too detailed and would complicate the model. They wanted to avoid it and did not immediately write it down as an object on the whiteboard. But the notion kept coming up during the subsequent process. Later, in discussing lanes, the team first identified that the notion of “Lane” is related to the notion of “length” in terms of the number of cars a road segment is able to hold between the two intersections. Toward the end of the design session, the designers also identified the notions of left-turn, right-turn, and straight-forward paths, related to lanes. In the end, the notion of “Lane” did not appear in the resultant model, but “the length of a road” and “left/right/straight paths” did.

4.3 Sharing mental imagery of software to be designed

We observed that the design teams brought up new concepts that were not part of a given requirement specification. Such concepts as “cop” and “faucet” explained in Observation-B seem to be the result of externalizing the designer’s mental imagery about the software to be designed [Petre 2009].

The Adobe team and the Intuit team demonstrated the effective use of such concepts as metaphors in understanding the complex behavior of an “intersection.” The Adobe team used the term “cop” to illustrate how an intersection handles cars flowing from one road to another. The Intuit team used the term “faucet” to understand the incoming flow of traffic into a road.
It is interesting to note that “cop” was made into an object in the Adobe team’s design, but “faucet” was not in the Intuit team’s design.

4.4 Dynamism of design process

The use of the DPS tool allowed us to observe and identify how the designers refine whiteboard drawings over the course of the design process (see Observation-J). These observations would inform whiteboard tool designers about how a whiteboarding tool could transcend the traditional ways of using a whiteboard.

We have found that the designers added labels to the drawn GUI window not as a GUI label component, but as names for the drawn objects so that they could talk about them. It was also observed that the two designers of the AmberPoint team sought more appropriate wording in listing object names, naming a portion of the user interface, and writing down a user story. One team changed a noun from singular to plural by adding an “s” to a word written on the whiteboard probably because it would correspond to a one-to-one or one-to-many relationship among the objects.

As shown in Observation-J, designers did not always keep consistencies among drawings on the whiteboard when changing names and labels. We also observed that there are temporal gaps between the time when the designers uttered a name and the time when the designers wrote it down (see Observation-H).

It was interesting to note that the Adobe team went into extreme detail in sketching GUIs on the whiteboard (see Observation-F). When the Adobe team sketched GUI windows in the study, they almost always enumerated all the possible elements and seldom omitted details. They did not seem to have minded redrawing different versions of the GUI ideas repeatedly (see Observation-F), similar to how an interaction designer repeatedly drew detailed sketches in a sketchbook [Nakakoji & Yamamoto 2010]. This team used the term "draw" almost as a synonym for "design" (see Observation-A).

The degree of detail may not necessarily be to complete the design detail but to invoke reflections. For instance, when Jim of the AmberPoint team started another attempt at drawing by saying, "So there's 120 seconds we want broken into increments" in specifying the duration of a cycle of a signal, he had come up with the idea of the "timeline" (AmberPoint: 00:37:26) representation.

At the same time, sketching a GUI window on the whiteboard does not afford the intended resolution for the designers. Jim of the AmberPoint team put "5 sec" as an increment and started putting dots in a timeline drawn in the popup window, but it became so small and congested that he changed to a 10-second increment. (AmberPoint: 00:37:50). It was not clear whether this was a design decision or just a way to accommodate his hand-drawing.

Reappropriation was observed for both graphical representations and textual representations (see Observation-E). We need more studies to distinguish reappropriation that is necessary by the nature of the design from reappropriation merely for the sake of convenience in saving time and effort to generate similar drawings.

4.5 Program design and interaction design

The AmberPoint team worked on the interaction design of the system, focusing on how a user would interact with the system through what user interface components in great detail. They started the task by drawing a GUI sketch for "laying out the roads" and a "visual map" (AmberPoint: 00:07:44), and then talked about what role this window would play in terms of what kind of functionality. For instance, "You need kind of a summary area to kind of tell you what your settings are for the individual intersections, and what kind of effect, like how much is the traffic backup at this light, or what's the average wait time at this light" (00:08:53).

This was a typical process of interaction design [Cooper 1999], which starts by focusing on how a user interacts with the system to be designed, and then leads to identifying necessary functionality. As such, the AmberPoint team detailed user interface components and walked through user interaction steps by interacting with a drawn GUI on a whiteboard through gestures

(Observation-A).

The Intuit team worked on the underlying substrate models of the simulation mechanism, primarily focusing on the program design aspect of software. Given the design prompt saying that the client wants to have "traffic simulation," the team spent a long time trying to understand how traffic operates in the real world, and discussing what assumptions to make to create a model that is simple to handle but covers the necessary functionality. Their struggle in dealing with "Lanes" of a "Road" (as explained in Observation-B) typifies the issue.

The Adobe team covered both aspects, ranging from simulation engines to user stories. They started by working on the object model, then specified control flows and rules, depicted user interface layouts and how a user interacts with the system, wrote user scenarios, outlined the main program structure, and drew ER diagrams. The level of detail they worked on was different from those of the other two teams, probably due to the limited amount of time assigned to the team.

The Adobe team demonstrated how the discussions of a user interface and those of a model were intertwined (see Observation-I). Thinking about controls over the model may trigger the designers to think about user interaction; conversely, thinking about user interactions through a user interface may trigger the designers to think about the model.

5. Final Thoughts

This chapter presents the list of our observations on what representations designers externalize and interact with during the early stages of software design. The representations we focused on include diagrams and text drawn on the whiteboard and uttered in the transcripts. In analyzing the data with DPS, we were particularly interested in when and how the representations were referred to, revisited, modified, and reappropriated over time.

By comparing the process of textual and graphical representations produced by the Adobe team and Intuit team, we found that the two teams seemed to be engaged in two completely different design tasks.

As discussed in the previous section, the Adobe team focused on the interaction design of the simulation software, and the Intuit team focused on the program design of its engine. We are aware of few studies that focus on how the interaction design (i.e., the design of the world of using) should be related to the program design (i.e., the design of the world of making). Software engineering research has traditionally looked into the design of program code. How a user would interact with the system, which is sometimes called experience design, has primarily been studied in the field of Human Computer Interaction (HCI).

Interaction design and program design are strongly tied together, as demonstrated by the Adobe team's process (Observation-I), but it is only so in the current practice. It is not clear whether the two design aspects should be done in parallel, whether one should precede the other, or whether they should be carried out by different teams of expertise or by a single team. More understanding is necessary to develop theories and models of the two aspects of software design.

We would like to further investigate qualitative and quantitative studies to identify the relationship between the design of the world of using and that of making.

Acknowledgments

We deeply appreciate the editors and the reviewers for valuable comments and encouragement.

References:

A. Baker, A. van der Hoek, Ideas, Subjects, and Cycles as Lenses for Understanding the Software Design Process, Design Studies, Vol. 31, Issue 6, pp. 590-613, November 2010.

H. Christiaans, R.A. Almendra, Accessing Decision-Making in Software Design, Design Studies, Vol. 31, Issue 6, pp.641-662, November 2010.

A. Cooper, The Inmates Are Running the Asylum: Why High-Tech Products Drive Us Crazy and How to Restore the Sanity. SAMS, Indianapolis, IN, 1999.

U. Dekel, J.D. Herbsleb, Notation and Representation in Collaborative Object-Oriented Design: An Observational Study, Proceedings of OOPSLA '07, ACM, New York, pp. 261-280, 2007.

E.Y-L. Do, Design Sketches and Sketch Design Tools, Knowledge-Based Systems Journal, Vol. 18, No. 8, pp. 383-405, 2005.

W. Ju, W.L. Neeley, T. Winograd, L. Leifer, Thinking with Erasable Ink: Ad-hoc Whiteboard Use in Collaborative Design. CDR Technical Report #20060928, Stanford University, 2006.

J. Kawakita, An Idea Development Method, Chuuko Shinsho, Chuuo Kouron-sha, 1968, Tokyo, Japan (in Japanese).

K. Nakakoji, Y. Yamamoto, N. Matsubara, Y. Shirai, Toward Unweaving Streams of Thought for Reflection in Early Stages of Software Design, IEEE Software, Special Issue on Studying Professional Software Design, Vol. 29, No. 1, pp. 34-38, January/February, 2012.

K. Nakakoji, Y. Yamamoto, A Study of Sketches for Interaction Design in the ARTware Project, Design Symposium 2010, JSPE, Tokyo, Japan, November 2010 (in Japanese).

M. Petre, Insights from Expert Software Design Practice, Proceedings of ESEC/FSE '09 pp. 233–242. ACM Press, New York, 2009.

M. Petre, A. van der Hoek, A. Baker, Editorial, Design Studies, Vol. 31, Issue 6, pp. 533-544, November 2010.

J. Rooksby, N. Ikeya, Collaboration in Formative Design: Working Together at a Whiteboard, IEEE Software, Special Issue on Studying Professional Software Design, Vol. 29, No. 1, pp. 56-60, January/February, 2012.

D.A. Schoen, The Reflective Practitioner: How Professionals Think in Action, Basic Books, New York, 1983.

M. Suwa, B. Tversky, Constructive Perception in Design, Proceedings of Computational and Cognitive Models of Creative Design V, Sydney, Australia, pp. 227-239, 2001.

J. Walny, S. Carpendale, N.J. Riche, G. Venolia, P. Fawcett, Visual Thinking in Action: Visualizations as Used on Whiteboards, IEEE Transactions on Visualization and Computer Graphics, Vol. 17, Issue 12, pp. 2508 - 2517, December, 2011.

Y. Yamamoto, K. Nakakoji, Interaction Design of Tools for Fostering Creativity in the Early Stages of Information Design, International Journal of Human-Computer Studies (IJHCS), Special Issue on Creativity, L. Candy, E. Edmonds (Eds.), Vol. 63, No. 4-5, pp. 513-535, October, 2005.