Follow that Sketch: Lifecycles of Diagrams and Sketches in Software Development

Jagoda Walny, Jonathan Haber, Marian Dörk, Jonathan Sillito and Sheelagh Carpendale
Department of Computer Science
University of Calgary
Calgary, Alberta
Email: {jkwalny, jmhaber, mdoerk, sillito, sheelagh}@ucalgary.ca

***Abstract*—Informal visualization in the form of sketching and diagramming has long been an established practise of professionals working in the fields of design, architecture, and engineering. Less is known, however, about the sketching and diagramming practices of computer scientists and software developers. Through a series of interviews with computer science researchers who develop software, we probed the purpose, contexts, and media in which they created and re-created sketches and diagrams, and the ways in which these informal visualizations evolved over time. Through our analysis we created visualizations of the observed sketching and diagramming lifecycles, which can contribute to a better understanding of the roles of sketching and diagramming in software development.**

I. Introduction

Visualization through sketching and diagramming plays an important role in the design process in various domains, including architecture, design, and engineering [6], [9], [19]. Such visualizations range from informal sketches to more formal diagrams following a standard visual language. They are used for everything from exploring, formation, and recording of ideas, to increasing one's understanding of concepts and communicating ideas to others. They also have considerable value within organizations–some engineers and architects save all sketches made during the course of a project as a record of all ideas and decisions made [6].

Although there has been much work in understanding the visualization practises of those designing physical objects (the aforementioned architects, designers, and engineers), it is unclear how much of this knowledge is transferable to software developers, who design digital objects. Software development differs from these domains in that it involves a combination of concrete visual elements such as user interfaces and more abstract elements like the structure of the source code.

In our experience, software developers create and work with a range of paper-based as well as digital visualizations of software concepts in their development work. These visualizations go beyond established visual formalisms such as UML, wireframes, or storyboards, and do not always use the available computer-supported tools, such as diagramming software or auto-generated UML diagrams. We have seen that software developers also draw their own informal visualizations, on various analog and digital media, that do not adhere strictly to these established formalisms.

To better understand how to provide tool support for integrating these visuals into developers' workflows we have conducted a qualitative study about the creation, use and transformation of diagrams, including transformations between paper and digital media. Rather than studying specific diagram types, we sought to gain an understanding of the various ways in which software developers currently use diagrams to develop ideas and solve problems. Based on qualitative interviews, we followed various sketches through their lifecycles involving a range of transitions and media types.

For this study, we recruited individuals who use sketches or diagrams for software design and development. These participants were not part of teams with formal practices in place that would influence how diagrams were to be used. Selecting these participants allowed us to study what types of sketching workflows software developers and researchers may tend towards naturally. By following individual sketches and diagrams through their lifecycles, we contribute a number of ways in which informal visualizations can have value beyond their initial creation (including as memory aids and for sharing with colleagues), and we characterize a range of actual sketch lifecycles that occurred in software development settings.

II. Diagrams and Sketches: Scope and definitions

We found that the visuals used by our study participants varied on a continuum from very sketchy to more refined (see Figure 1). One end of this continuum is similar in spirit to Buxton's [3] characterization of sketches as: appearing to be made quickly; disposable; plentiful; following conventions that distinguish them as sketches; having a fluidity evoking "a sense of openness and freedom"; containing minimal detail; being no more refined than the certainty in the creator's mind; "suggesting" rather than "telling"; and being intentionally ambiguous. The other end is characterized by formal drawings with crisp lines, often following some predefined visual language or conventions.

Fig. 1. Sketches and diagrams can be situated along a formality spectrum according to their sketchy vs rigid nature.

We were interested in both sketches and diagrams as visual artifacts. The visual artifacts we saw can be arranged on our spectrum from *informal* to *formal* representations (see Figure 1). Throughout this paper, we call those visual artifacts towards the informal end of the spectrum *sketches*, and those visual artifacts towards the formal end of the spectrum *diagrams*. Some ambiguity occurs in the middle of the spectrum, where formal diagram elements are mixed with sketchy elements. We saw both sketches and diagrams that were created ad-hoc (rather than strictly following some software visualization convention such as UML); we call these *informal software visualizations*.

To better understand how software developers use sketches and diagrams in their work, we focus on two key aspects of their informal visualization practices: the contexts and tools used in the creation and use of sketches and diagrams (the how); and the reasons behind the evolution of these visual artifacts (the why). In the remainder of this paper we describe work related to the use and value of sketches in various domains, explain our study methodology, then present how sketches and diagrams used in software development transition through their lifecycles in terms of purpose, medium, and context. Before concluding the paper, we discuss the variety and uses of informal visualizations we encountered.

III. Related Work

Informal visualizations such as sketches have long been used to share, clarify, communicate, and store thoughts and ideas [22]. Due to their ability to support thought processes and idea development [17], [20], they have been identified as particularly valuable for creative design tasks [7], [11], [14]. Research in this area has been especially active across the fields of design (particularly architectural design) [8], [16] and engineering [6].

The informality of sketches is what makes them so valuable: the idea generation process is supported by their inherent ambiguities and simplifications. Tversky [21] notes that experienced designers have strategies for leveraging the ambiguity of their sketches to generate new ideas in a process called constructive perception. In software engineering, many have recognized that rough, low-fidelity, or "sketchy" prototypes [14], [15], [23] are more effective at generating ideas and encouraging feedback in the early stages of software design than their polished counterparts. In our research we are interested in the transformations that these early, informal sketches undergo in a software development setting.

Interest in digital sketching and diagramming tools dates back to 1963 with Sutherland's Sketchpad [18] and remains an active area of research [12]. Despite this interest and recent advances in pen- and touch-enabled computing, many still prefer paper for their sketches [13]. We argue that this less than complete adoption of software tools may be because in order to fully support digital sketching, these tools must take into account the entire lifecycle of these sketches.

Making a sketch is not an isolated activity; a series of sketches is like a conversation in which the sketch is just as instrumental as the sketcher in pushing an idea forward (what Goldschmidt [10] calls the "backtalk" of sketches). We extend this idea of a sketch conversation and demonstrate that a sketch has a lifecycle: it starts out as an idea and may end up as far as having a formal, polished representation. This idea was inspired in part by Cherubini et al.'s observation [4] that some sketches and diagrams in a software development setting were *reiterated* - revisited over and over again.

In studying sketch lifecycles we inevitably need to consider transitions between different media. There is some interest in integrating sketches across various media, for example, Branham et al.'s Reboard system [2] captures whiteboard drawings and makes them available for reuse across devices, or Brandl et al.'s work on integrating paper-based and digital interfaces using paper enhanced with Anoto patterns [1]. Our study is intended to expand our understanding of such transitions by providing a snapshot of how and why a software developer's sketch might transform across media and contexts.

IV. Methodology

Our study involved semi-structured interviews with eight participants who explained the lifecycles of one or two sketches and diagrams as part of their software development.

Participants. We recruited software developers who used visualizations such as sketches or diagrams related to their software development practice. We recruited participants via a mailing list reaching graduate students and professors in a computer science department at a university. Our participants included six graduate students, one university professor, and one postdoctoral fellow. They were not part of teams that had formal practices dictating how sketches and diagrams were to be used; they all had relative freedom to design their own workflows. While participants made diagrams as and when they wished, they were sometimes influenced by their supervisor's wishes or advice. This allowed us to see the types of workflows our participants tend towards naturally.

Study Design. We conducted our study in two phases. In the first phase, we interviewed five participants about the lifecycles of one or two of their sketches or diagrams that related to software development. These five interviews were semi-structured and lasted approximately one hour each. After initial open-coding analysis [5] of the data collected in phase one, we conducted a second round of interviews with four additional participants lasting approximately thirty to forty-five minutes, and were structured similarly to the interviews in phase one but with an increased focus on transitions in sketching practices. We discarded one interview due to the participant not being involved in any software development.

Semi-Structured Interviews. At the beginning of each interview the participants filled out a questionnaire to establish some background information about the software development project, their role in it, and their previous experience with sketching or diagramming. During the interviews, we asked questions about the context in which the sketch was created, characteristics of previous versions, the purpose of creating the sketch, the roles of people involved in creating or viewing

the sketch, the tools and techniques used, and their personal experience during the creation of sketches. In the interviews of the second phase, we specifically asked participants to show us an example of a sketch they actively revisited. We asked questions about this sketch and their typical behaviours.

Analysis. All interviews were videotaped to record sketches in context of participants' gestures and statements during the interview. We transcribed the audio as well as the relevant diagram characteristics. After phase one, we performed an open coding of the transcript data and then determined what interesting observations emerged from the data in the form of broad categories. From this analysis it emerged that, in order to study the lifecycles of these sketches and diagrams, we would need to focus on transitions between sketch stages and media.

We then conducted phase two of our interviews with an increased focus on these transitions. We analyzed phase two data and re-analyzed phase one data in light of transitions. Based on this open coding and our observations of the video transcripts we created diagrams depicting the lifecycles of the sketches and diagrams revealed by the participants.

V. What is a Transition?

From our analysis it emerged that the clearest way to describe the visualization lifecycles we saw was through the lens of the transitions they passed through.

We define a transition as a passage of a visualization from one state to another, where a state describes medium, context (personal or group), actual form the diagram takes (thus an alteration, annotation, or re-creation would all be a change of state), or a change in use status (actively being used, not in use). This definition emerged both from answers from our participants regarding the contexts in which they created sketches or diagrams and from our analysis of the data.

Each transition can be grouped into one of five categories:

- *Creation*: A special-case transition in which a visualization goes from existing as an idea in the developer's mind to having an actual form on an analog or digital medium.
- *Iteration*: This transition involves a change in the form of the diagram, either through redrawing, annotating, or summarizing of previous diagrams.
- *Copying*: This transition involves making a direct reproduction of the diagram through such means as copying, scanning, or taking a picture.
- *Archival*: This transition involves a change in status of the diagram to inactive. The diagram is stored somewhere, but not actively accessed.
- *Discarding*: This transition involves discarding the diagram, usually deliberately.

Together, these transitions form a sketch and diagram visualization lifecycle. In the next section, we describe the range of lifecycles we observed through the lens of these transitions.

VI. Diagrams/Sketching practices

We use a series of lifecycle diagrams to show both the variation and the commonalities between the sketches and diagrams we followed. The sketching and diagramming practices from our analysis are represented in Figures 3–5. The horizontal axis is a non-linear representation of time, while the vertical axis indicates the relative number of people involved in the creation, manipulation, or discussion around the media object. The lowest position along the vertical axis indicates single-person use, and higher positions indicate more people are exposed to the sketch. Icons represent the type of medium used; icon labels indicate the action taken with the medium; arrows represent transitions between media types and/or states of the diagram or sketch; arrow labels represent reasons for transition; and varying colours and arrow types indicate different types of transitions. The media and transition types are listed in Figure 2.

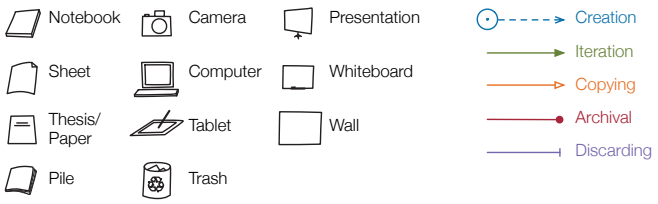

Fig. 2. Media and transition types encountered in interviews; this is a legend for Figures 3 through 5.

Each participant description characterizes the participant regarding their sketching practice and describes a sketch workflow (represented in the diagrams) along multiple transitions ranging from creation, iteration, and copying to archiving and discarding. We grouped participants into three classes of sketching practices: those with a thoroughly considered workflow (Section VI-A), those with a freeform workflow (Section VI-B), and those with low sketching activity (Section VI-C).

A. Participants with thoroughly considered workflows

Participants in this category put a high amount of effort into thinking about and adjusting their workflow to support sketching and diagramming.

Participant #4 (P4), unlike any other participant, took great care to set up a system in which all of his sketches and diagrams would end up in digital form in an efficient manner. He was the only participant we interviewed who regularly and deliberately used a tablet PC to create sketches and diagrams. (P1 and P3 both owned tablet PCs but felt that they were too cumbersome to use as compared to paper or whiteboards.)

P4 was aware of the limitations of tablet PCs but was determined to create a workflow that worked for him. He scanned the most valued of his old sketchbooks into a digital archive and carefully chose a model of tablet PC with a weight and responsiveness he was comfortable with. He also carefully chose his diagramming software, a vector-based drawing application that specifically supports tablet PCs, allows the creation of custom templates, and exports to vector (PDF) and JPEG formats. Because P4 spent so much time optimizing his diagramming system, he was able to explain to us in detail his typical diagram lifecycle (see Figure 3).

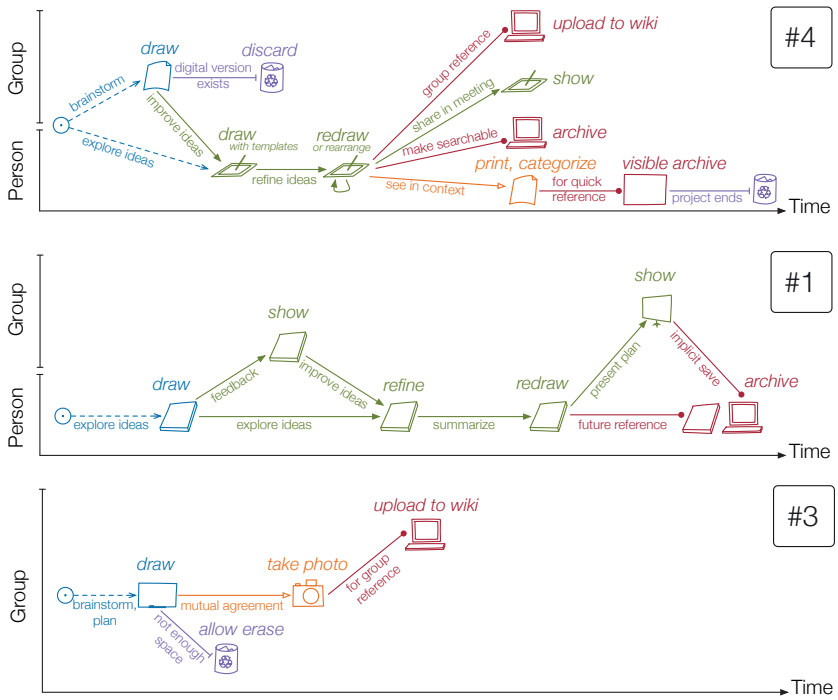

Fig. 3. Depictions of diagram lifecycles by participants with thoroughly considered workflows (Participants #4, #1, and #3). Participant #4's lifecycle showcases extensive, deliberate use of digital media; Participant #1's shows iterative refining of prototype sketches and diagrams for a user interface; and Participant #3's depicts the cycle of a diagram-turned-contract.

Creation. P4 engages in two types of brainstorming: individual and in groups. When exploring ideas alone, he draws directly on his tablet PC, using his chosen drawing program. He does this also in public (e.g., on a bus), but admits to feeling self-conscious using a computer instead of paper out in public. When brainstorming in groups, P4 usually uses paper, often large sheets everyone can draw on. In both cases, P4 often references previous sketches when exploring new ideas.

Iteration. After the creation stage, P4 translates ideas captured on paper to his tablet PC. Refining his ideas, he rearranges or redraws diagrams on his tablet PC. He cites the ability to make templates and rearrange elements as the biggest advantage of the tablet PC over paper.

Copying. To categorize his diagrams and see them in context, P4 prints out his diagrams (in colour) and places them in groups on the wall in his workspace organized by project. We can call this a "visible archive" of diagrams. P4 references this archive when exploring new ideas, beginning the diagram lifecycle again.

Archiving. Once the digital diagrams have matured through iteration, several things happen: P4 uploads the diagrams to a wiki accessible to his research group; the diagrams are saved to a digital tablet device for sharing during meetings; additionally, they are uploaded to a searchable online archive that supports optical character recognition making it easy to find specific diagrams or browse through old diagrams.

Discarding. P4 actively discards paper versions of diagrams that have digital equivalents, except for those he drew in sketchbooks before setting up this digital workflow. The sketches that were made as part of a group brainstorming are discarded as soon as digital versions exist. When a project ends, he discards the corresponding printed diagrams on the wall because he has no further need to see them in context.

Participant #1 (P1) was designing a user interface for a system used in a professional setting. This was a research project but, as in business projects, she had to satisfy the users and the decision-makers. Her sketching workflow was not as carefully considered as P4's, but she was reflective about it due to the encouragement from her supervisor to include sketching in her work (see Figure 3). She owned a tablet PC, but mainly used her sketchbook for sketching initial ideas.

Creation. P1 would sketch ideas in her free time, including small pockets of time such as when she was waiting to pick up her kids from school. Her main medium for brainstorming was a sketchbook; a tablet PC was too cumbersome and took too long to boot up.

Iteration. P1 would share selected sketches with her target user base directly from the sketchbook. As the ideas matured, she summarized them in her notebook (drawing a "summary sketch"), redrew them in a digital format and placed them in a presentation, which she gave to a group of decision makers. While she would share formal diagrams as part of presentations to decision makers, she would show more casual sketches to a smaller group of the user base earlier in the process.

Copying. P1 did not mention any scanning or photographing of her sketches, though she did reproduce her diagrams into PowerPoint when necessary for a presentation.

Archiving. P1's main way of archiving sketches and diagrams was her sketchbook, in particular the summary sketches.

Discarding. P1 did not discard her sketchbooks since they served as possible future reference sources. She also did not delete digital sketches that would be implicitly saved.

Participant #3 (P3) has a supervisory role and does not actually create software himself, but his students do, following a method based on agile software development methodology. The diagrams he uses are drawn by and/or with his students during group and project meetings (see Figure 3).

Creation. P3 and his students would brainstorm and plan by drawing sketches and diagrams on a whiteboard.

Iteration. P3 did not modify the sketches his students drew; at most he would reference them.

Copying. If a sketch or diagram represented a mutual agreement of tasks to be completed, P3 or his student took a photo and uploaded it to a wiki that all of P3's students had access to. This was considered to be evidence of a "contract" between student and supervisor.

Archiving. Sketches or diagrams were left on the whiteboard until the space was needed for a different use. Photos were archived on a wiki accessible to the entire group.

Discarding. P3 considers whiteboard sketches to be transient in that they could be erased any time.

B. Participants with a freeform workflow

Participants in this category sketched often and had their own naturally emerging workflows for dealing with sketches and diagrams, which they adapted in collaborative settings when necessary.

Participant #5 (P5) programs graphical computer software, so her work is very visually oriented. She described to us the lifecycle of a diagram used for debugging (see Figure 4).

Creation. Initially, P5 was debugging her software using a debugger, but could not figure out a particular problem. She began sketching out the problem on the most readily available drawing surface at the time - a loose piece of paper from a stack of loose paper at her desk that she keeps for this purpose.

Iteration. When the paper sketch did not solve P5's problem she felt she needed to "clear her head" and "change her environment", so she moved to a communal whiteboard and began to sketch out her problem. Here, she discovered that she needed to account for several cases, and was missing one. Next she moved back to her desk and began to plan out how to implement the missing case on a piece of paper. The piece of paper stayed in a pile on her desk serving as a memory aid, so that when she switched tasks away from the implementation and then back, she could go back to the diagram and remember the state of mind she was in when she drew it.

Copying. P5 later decided that the drawing on the whiteboard did quite a good job of explaining the problem she was working on and the solutions, so she took a photo of it, which she then transferred to her hard drive.

At a later date, she planned to reference this photo while creating a formal, explanatory diagram that she would publish while documenting her software in a thesis or conference publication. She acknowledged that the final diagram might have slightly different elements, but was confident that the whiteboard sketch provided a good basis.

Archiving. The piece of paper stayed in a pile on P5's desk serving as an easily accessible memory aid.

Discarding. She said she would discard it when it no longer held any value for her as a memory aid, and was instead simply cluttering up her desk.

Participant #6 (P6), like P5, also programmed graphical computer software (see Figure 4). He showed us two different types of typical (for him) sketches. The first was a type of drawing he often used for debugging; the second, a typical diagram used to explain a concept to someone else.

Creation. When using a debugger, at times P6 finds that the numbers shown by the debugger (coordinates) provide little meaning without a visual reference. So he draws them on a piece of paper along with a visual representation of the coordinates to help solve the problems.

The second diagram type, the explanation sketch, is drawn in a meeting with someone else and is intended to be a transient diagram - used only for explanation and then erased.

Iteration. A debugging diagram is specific to a particular part of P6's software, so it has value to him when he returns to work on that part of this source code. He saves these diagrams in a notebook.

The explanation sketch turned out to be a type of diagram he drew often, so P6 used it to create a more generalized diagram that explained the underlying concept. He did this in digital format, and kept this generalized diagram for presentations or publications.

Copying. When debugging his code P6 occasionally needs a diagram that includes a screenshot of his software. In these cases he creates the diagram digitally rather than on paper.

For P6, an explanation sketch at times turns out to represent a particularly useful way of organizing or explaining the concept, in which case he takes a photo of the diagram and later creates a refined, formalized version of it for sharing in presentations or publications.

Archiving. The debugging diagrams have value to P6 when he returns to work on that part of this source code, so he saves these diagrams in a notebook. The explanation sketches, on the other hand, would only be saved indefinitely on his hard drive if they stand out in terms of clarity of explanation.

Discarding. The explanation sketches that are "rough or simple" are used only once and then get discarded or erased, unless determined to be useful in some way for the future.

Participant #2 (P2) is a project lead on a research project, working with one developer, and has a supervisor. P2 has an art background and draws digitally extremely infrequently. She states she would have drawn diagrams by hand in her thesis if that were possible. Her workflow is very freeform–sketches are drawn on loose paper and archived in piles. She showed us two visual artifacts: a network architecture diagram and a user interface diagram.

Creation. The first artifact (represented in Fig. 4) was a network architecture diagram. In order to plan out how several

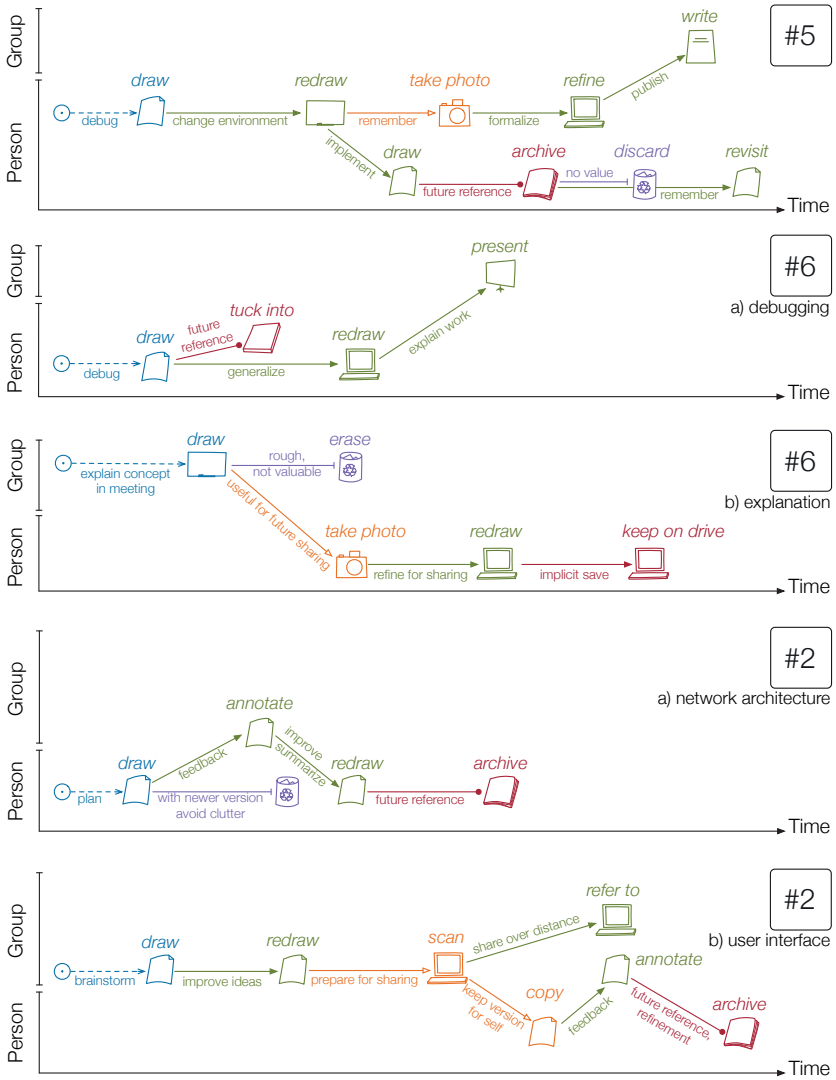

Fig. 4. Depiction of diagram lifecycles for participants with a freeform workflow (Participants #5, #6, and #2). Participant #5's lifecycle shows a debugging process; Participant #6 has two lifecycles, one for debugging (a), and one for explanation of a concept (b); Participant #2 also has two lifecycles, one representing her network sketches (a) and one representing her interface sketches (b).

servers would communicate with each other, she drew an initial sketch on a piece of paper.

The interface diagram was created on paper as she and a colleague brainstormed different interface ideas.

Iteration. P2 showed the network architecture sketch to a colleague to receive feedback and annotated the sketch with his comments. Once her vision of the network architecture had matured, she drew a diagram that summarized her plan, similar to P1's "summary sketches". She kept this summary diagram in a pile of loose papers connected to the project and referred to it when necessary.

For the interface sketch, she worked with her colleague to iteratively improve the ideas, drawing new sketches when necessary.

Copying. Once satisfied about the number of ideas, they needed to share them with their supervisor, who was away at the time. This was done by scanning and emailing the paper prototypes to their supervisor.

P2 wanted her collaborator to have the original prototype sketches so that he could reference the correct colours; P2 still wanted a copy for herself, however, so she made photocopies for her own use. She brought these photocopies to the meeting with her supervisor and annotated them with feedback from the meeting.

Archiving. P2 kept several iterations of the interface diagram, particularly the photocopies she had annotated at her meeting with her supervisor. She kept these photocopies with her other loose papers and referenced and refined them as necessary.

Discarding. P2 discarded previous versions of the network diagram to avoid clutter from material that was no longer up to date.

C. Participants with low sketching activity

Participants in this category used diagrams more so than sketches and did not place a high importance on thinking about their sketching workflow.

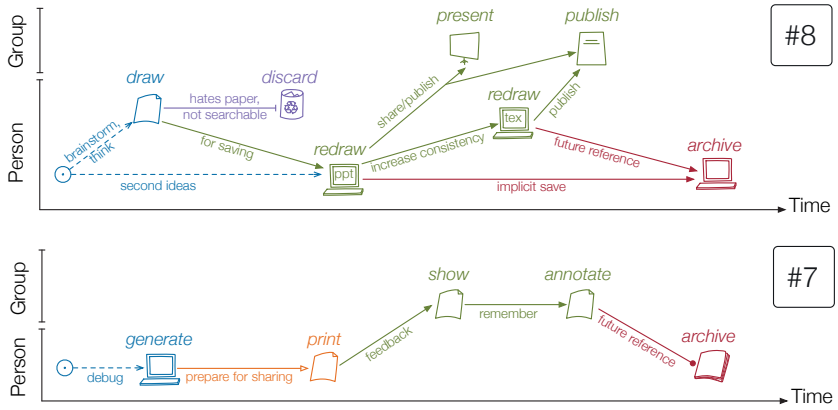

Fig. 5. Depictions of diagram lifecycles from participants with low sketching activity (Participants #8 and #7). Like Participant #4's workflow, Participant #8's workflow is centred around digital media. Participant #7's lifecycle shows an automatically generated diagram turned into an annotated printout.

Participant #8 (P8), who was creating system architectures for security, used paper but expressed a strong dislike for it. In contrast to P4, who actually liked paper but reduced its use in favour of the affordances of digital diagrams, P8 continued to use paper, but immediately created digital versions of valuable sketches and discarded the paper. He disliked that paper was not searchable, and he also disliked the clutter it created. P8 walked us through several sample diagrams (see Figure 5).

Creation. P8 still used paper for the majority of his brainstorming. In some cases he would insert mature ideas directly into slideshow software with basic diagramming tools. It was quite easy for him to create diagrams in the slideshow software, so he would put both personal ideas and diagrams intended for presentation in it.

Iteration. If a diagram was important enough to save it, P8 would create a digital version of it as soon as possible–either in the slideshow program or in the typesetting environment LaTeX. In some cases for formal presentation he kept the slideshow diagrams; in other cases he moved them into LaTeX. The slideshow program allowed him to "move boxes and lines around", but LaTeX, which generates a diagram from a plain text representation, let him organize diagrammatic elements conceptually but with less flexibility in layout.

P8 had two main reasons for using LaTeX diagrams, which took several hours to create: LaTeX had superior support for mathematical equations, and it was the format in which he created his publications, so making his diagrams in it provided a visual consistency he sought.

Copying. P8 did not take photos, print, or scan, but his LaTeX diagrams were fairly faithful representations of the original diagrams.

Archiving. P8, like all of our participants, did not explicitly delete digital files. Digital diagrams tend to be implicitly archived indefinitely.

Discarding. Pieces of paper with sketches were immediately discarded after they had been redrawn digitally.

Participant #7 (P7) is a researcher who was working on a network protocol when he told us about his sketching practices. His goal was to create a protocol that optimized particular parameters. In contrast to most participants' sketching practices moving from physical to digital media, the sketch lifecycle of P7 starts out with a digital diagram transitioning to annotated printouts.

Creation. In order to judge the performance of his protocol, P7 set up an automated method of generating graphs from log files produced by his software.

Copying. He prints out these graphs, staples them together, and together with his supervisor, annotates the diagrams with various comments and ideas for improvement.

Iteration. The process of generating these graphs is streamlined and he can re-generate any graph when needed. However, he has not yet had to re-generate any of the diagrams, and he frequently makes use of the printed versions because they contain many valuable annotations.

Archiving. He stores each stapled group of graphs in a neat pile on his desk (newest diagrams on top), which he can reference as he works on his protocol. The pile on his desk contained all of the graphs he had generated throughout the whole project.

Discarding. Up to the point of the interview, P7 had not discarded any diagrams.

VII. Discussion

As we followed our participants' sketches and diagrams through their lifecycles, we saw that these lifecycles are quite complex. Initial sketches and diagrams undergo a variety of transitions. Our participants created and worked with sketches and diagrams for a wide range of purposes related to software development: generating, refining, and summarizing design ideas, evaluating software performance, debugging and for explanation and mutual agreement with colleagues. These visuals were created and recreated on various media including paper, notebooks, printers, cameras, photocopiers, Tablet PCs, PCs, and using auto-generators. Their contexts ranged from informal to formal, individual to group. The visuals served many uses including ideation, communication, and distillation.

Ideation. Sketches are frequently discussed as great vehicles for an ideation process [7], [9], [12], [20], and generating many sketches is a recommended technique for finding good ideas [3]. We found that while some sketches were only transient (such as those created on shared whiteboards and never

saved), many sketches were saved, iterated upon, reused and transformed. These sketches were valued as idea generators not only during creation, but also later in the process. For instance, it was noted that re-looking at an earlier sketch could trigger new and different ideas.

For most of our participants, the transitions within the ideation process (mainly creation and iteration transitions) happened almost exclusively on analog media. These transitions are a throwback to Goldschmidt's [10] idea of a sketch as a conversation, and are where constructive perception [20] is likely to occur due to the sketches' informality and ambiguity. It was in these stages that our participants had the most variation in, and were most passionate about, the tools they used.

Communication. Much of the sketch lifecycle was influenced by changing communication needs. People redrew sketches to clarify aspects for others. They annotated each others' sketches to record agreements and variations in ideas that arose during discussions. Sketches were transitioned into different media to better serve given communication needs.

Distillation. Transitions that were not through a direct copying process such as photos, scans and prints, were used to both refine and clarify ideas and to open up discussions about selected aspects again in more formal sharing settings.

Each transition type was associated with different benefits. Creation translated internal ideas to external visuals. Iteration transitions, which made up the bulk of the lifecycles we saw, were usually for idea generation, feedback, or for formalizing. Copy transitions tended to be for keeping analog visuals as a memory aid or for switching to a shareable medium. Archive transitions tended to be implicit (a file left on a hard drive) unless deliberately archiving for sharing with a group. Discard transitions were more frequently analog and tended to be explicit, usually to reduce clutter.

While this study has revealed a great deal of variety in sketch and diagram lifecycles, the full extent of the variety is not yet known. More variations may emerge in different software development teams as people adapt to various workflow constraints. However, it is clear that the informal visuals used in software development have value beyond initial creation as they are modified, copied, shared, and archived.

Needed are visualization tools that take into account that developers have these varied sketch lifecycles. Such visualizations might flexibly support iteration transitions and archiving of particularly valued visualization instances if they are meant for idea generation purposes or debugging; transition between various media and contexts if they need to be shared or formalized; or transition through various levels of formality if they are meant to be used in several different contexts.

VIII. Conclusion

We have provided a snapshot of the lifecycles of sketches used by eight people during their software development processes. Although this is just a small sample of software developers, the complexity and individuality in these lifecycles has considerably expanded our understanding of these processes. The richness of our findings can be seen in the subtle differences and unique evolutions of the sketching and diagrammatic life cycles observed.

Informal visuals clearly factor into the workflows of software developers, and some of these go on to have value within a larger group or formal setting. Effective tool support for integrating sketching into the software development workflow, regardless of medium, must consider not just transitions across media but transitions across purpose and context to support a broad range of work styles.

References

[1] P. Brandl, M. Haller, J. Oberngruber, and C. Schafleitner. Bridging the gap between real printouts and digital whiteboard. In *Proc. of AVI*, pages 31–38. ACM, 2008.

[2] S. Branham, G. Golovchinsky, S. Carter, and J. T. Biehl. Let's go from the whiteboard: Supporting transitions in work through whiteboard capture and reuse. In *Proc. of SIGCHI*, pages 75–84. ACM, 2010.

[3] B. Buxton. *Sketching user experiences: Getting the design right and the right design*. Morgan Kaufmann, 2007.

[4] M. Cherubini, G. Venolia, R. DeLine, and A. J. Ko. Let's go to the whiteboard: How and why software developers use drawings. In *Proc. of SIGCHI*, page 566. ACM, 2007.

[5] J. Corbin and A. Strauss. *Basics of Qualitative Research*. SAGE Publications, 3rd edition, 2008.

[6] E. S. Ferguson. *Engineering and the Mind's Eye*. The MIT Press, 1994.

[7] V. Goel. *Sketches of thought*. The MIT Press, 1995.

[8] G. Goldschmidt. The dialectics of sketching. *Creativity research journal*, 4(2):123–143, 1991.

[9] G. Goldschmidt. On visual design thinking: The vis kids of architecture. *Design studies*, 15(2):158–174, 1994.

[10] G. Goldschmidt. The backtalk of self-generated sketches. *Design Issues*, 19(1):72–88, 2003.

[11] M. D. Gross and E. Y. L. Do. Ambiguous intentions: A paper-like interface for creative design. In *Proc. of UIST*, pages 183–192. ACM, 1996.

[12] M. Haller, P. Brandl, D. Leithinger, J. Leitner, T. Seifried, and M. Billinghurst. Shared design space: Sketching ideas using digital pens and a large augmented tabletop setup. *Advances in Artificial Reality and Tele-Existence*, pages 185–196, 2006.

[13] G. Johnson, M. D. Gross, J. Hong, and E. Yi-Luen Do. Computational support for sketching in design: A review. *Foundations and Trends in Human-Computer Interaction*, 2(1):1–93, 2009.

[14] J. A. Landay. *Interactive Sketching for the Early Stages of User Interface Design*. PhD thesis, Pittsburgh, PA.

[15] M. Rettig. Prototyping for tiny fingers. *Commun. ACM*, 37:21–27, 4 1994.

[16] P. A. Rodgers, G. Green, and A. McGown. Using concept sketches to track design progress. *Design Studies*, 21(5):451–464, 2000.

[17] M. Schütze, P. Sachse, and A. Römer. Support value of sketching in the design process. *Research in Engineering Design*, 14(2):89–97, 2003.

[18] I. E. Sutherland. Sketchpad: A man-machine graphical communication system. In *Proc. of AFIPS Spring Joint Comp. Conf*, volume 23, pages 329–346. Citeseer, 1963.

[19] M. Suwa and B. Tversky. What do architects and students perceive in their design sketches? A protocol analysis. *Design Studies*, 18(4):385–403, 1997.

[20] B. Tversky. What do sketches say about thinking? In *2002 AAAI Spring Symposium, Sketch Understanding Workshop, Stanford University, AAAI Technical Report SS-02-08*, 2002.

[21] B. Tversky, M. Suwa, M. Agrawala, J. Heiser, C. Stolte, P. Hanrahan, D. Phan, J. Klingner, M. P. Daniel, and P. Lee. Sketches for design and design of sketches. *Human Behaviour in Design: Individuals, Teams, Tools*, page 79, 2003.

[22] D. G. Ullman, S. Wood, and D. Craig. The importance of drawing in the mechanical design process. *Computers & graphics*, 14(2):263–274, 1990.

[23] Y. Y. Wong. Rough and ready prototypes: Lessons from graphic design. In *Posters and short talks of SIGCHI*, page 84. ACM, 1992.