Supporting Informal Design with Interactive Whiteboards

Nicolas Mangano[1], Thomas D. LaToza[1], Marian Petre[2], André van der Hoek[1]

[1] Department of Informatics
University of California, Irvine
{nmangano, tlatoza, andre}@ics.uci.edu

[2] Centre for Research in Computing
The Open University
m.petre@open.ac.uk

ABSTRACT
Whiteboards serve an important role in supporting informal design, providing a fluid and flexible medium for collaborative design. Interactive whiteboards offer the potential for enhanced support for manipulating content, managing sketches, and distributed work, but little is known about how this support affects the practice of informal design. To understand the opportunities and challenges, we first conducted a literature review, identifying 14 behaviors that occur during informal design. We then designed an interactive whiteboard system to support all of these behaviors and deployed the system to three groups of designers. Through usage logs and interviews, we examined the effects of interactivity on whiteboard use across a wide spectrum of design behaviors, identifying ways in which interactive whiteboards support the practices used in physical whiteboards and where they enable designers to work more effectively.

Author Keywords
Design; sketching; interactive whiteboard; informal design

ACM Classification Keywords
D.2.2 [**Software Engineering**]: Design Tools and Techniques – *Computer-aided software engineering (CASE)*.

INTRODUCTION
Interaction designers and software developers generating and refining ideas engage in informal software design, turning to the whiteboard rather than tools for formal notations for the flexibility and fluidity it provides [6]. Yet while designers wish to manipulate content in more sophisticated ways than adding and erasing strokes [11], physical whiteboards remain a passive medium lacking active support for design. In response, nearly three decades of research [31, 18] has explored the design of interactive whiteboards, investigating approaches for sketch recognition [5, 21, 16, 9, 7], sketch management [31, 32, 26, 25, 13, 19, 3, 12], and distributed sketching [19, 14, 15, 29]. Yet interactive whiteboards are not widely used in practice [17].

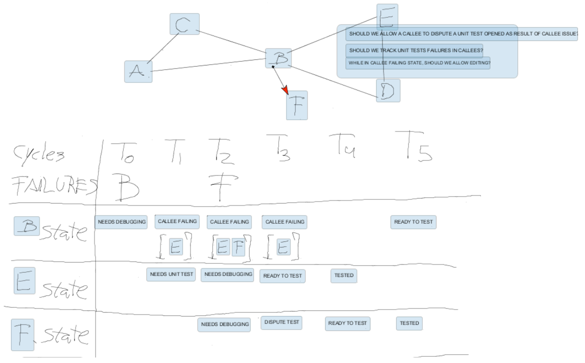


Figure 1. A sketch a designer drew with Calico.

We set out to understand the opportunities and challenges that interactive whiteboards afford in supporting informal software design. What behaviors are important for an interactive whiteboard to support to provide increased utility? How can interactive whiteboards effectively support these behaviors? How does supporting these behaviors impact the practice of informal design? What challenges remain inherent in the medium afforded by interactive whiteboards?

We first conducted a review of the software design literature, identifying 14 behaviors important to support in informal design. We then designed a single unified tool – Calico – intended to preserve the fluidity and flexibility of the whiteboard while more effectively supporting the full range of sketching, navigating, and collaboration behaviors we identified. Finally, we conducted a field deployment of Calico to three groups of designers, recording usage logs and interviewing designers about their experiences.

Our results illustrate the breadth and diversity of informal design at the whiteboard. Designers used Calico to create a wide range of sketches (e.g., Figure 1). The contexts in which designers worked – the nature of the design problems they faced, whether they were collocated or distributed – led to different usage of the features provided. A key benefit of interactive whiteboards was the infinite number of canvases they can provide, allowing designers to consider more alternatives and maintain a record of their design. Enabling designers to express relationships between canvases allowed designers to consider their design at a meta-level, providing context with which to interpret and reconstruct past designs. Our results also identified behaviors that are important to more effectively support, such as juxtaposing sketches and identifying marks in collaborative settings.

CHI 2014, April 26 - May 01 2014, Toronto, ON, Canada
Copyright 2014 ACM 978-1-4503-2473-1/14/04···$15.00.
http://dx.doi.org/10.1145/2556288.2557411

Previous work presented an earlier version of Calico [23]. This paper presents a system redesigned from scratch to support not 4 but 14 distinct design behaviors (including distributed sketching) and a field deployment of its use. Other work has examined its use in the classroom [22].

RELATED WORK

Decades of research into interactive whiteboards has explored a variety of approaches, including sketch recognition, sketch management, and support for distributed sketching (see Johnson et. al. [18] for a review). Other work has focused on understanding the use of groupware on large displays – including for distributed sketching – in practice.

Sketch recognition systems interpret a user's strokes, translating them into a formal object. Early systems used a predefined formal notation for interpreting sketches, such as UML diagrams [5] or user interface mockups [21], using the rules of the notation to provide feedback. Later systems explored user-expandable notations [16] and increased flexibility by delaying interpretation until desired [9], sometimes even while retaining a sketchy appearance [7].

Many systems have explored support for managing the many and varied sketched artifacts that are produced during meetings. Early approaches organized sketches using a filmstrip [31], hyperlinks [32], or hierarchical perspectives [26]. Later work automated particular aspects of managing sketches by automatically grouping clusters of sketches in close spatial proximity [25], shrinking sketches when moved to the periphery [13], or using metaphors such as Post-It Notes to organize and relate sketches [19]. Other systems capture and present the history of interactions with a whiteboard as a tree of storyboards [3] and allow designers to navigate a network visualization of canvases [12].

Several systems have also explored techniques for supporting synchronous and asynchronous design amongst collocated and distributed designers. Tele-board [14] is a distributed whiteboard and Post-It Note tool that allows designers to generate sticky notes from remote locations, group them, and review whiteboards in a history viewer. Designer's Outpost [19] helps communicate distributed designers' gestures and body language using shadows on the whiteboard. Team Storm [15] allows designers to sketch in either private spaces or public spaces, allowing designers to interact with and provide feedback on others' sketches. Gambit [29] allows designers to use a variety of devices together including large displays, laptops, tabletops, and phones.

A few studies have investigated the impact of groupware systems for supporting design with large format displays on practice. A field deployment of Tele-Board [14] – using traditional computers rather than an interactive whiteboard – found that moving between synchronous and asynchronous modes of work allowed designers to use the system to prepare for meetings and saved time during meetings, as designers did not need to wait to sketch their ideas. Another study [17] examined the use in practice of several large-display groupware systems for informal collaboration, communication, and awareness. The study suggested the importance of supporting critical tasks, making the system's value evident, supporting a breadth of collaboration practices, deployment in visible ways, low barriers to use, and having a core group champion the system. Our work builds on these studies, focusing specifically on the impact of interactive whiteboards on informal design.

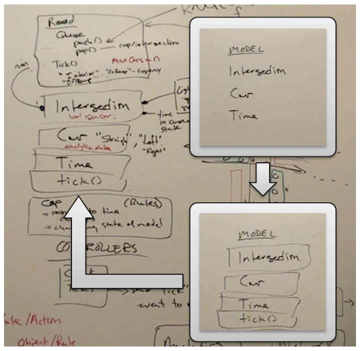


Figure 2. Designers evolve sketches created using simpler notations (list) into richer notations (class diagram).

DESIGN BEHAVIORS

We reviewed the software design literature and identified 14 behaviors that occur during design at the whiteboard.

How Designers Sketch

Designers draw different kinds of diagrams. To explore a design problem, software designers sketch many different types of diagrams, often within the same canvas [1, 8], enabling designers to explore an issue from different angles.

Designers draw what they need, and no more. Few sketches are created with extensive detail; rather, designers create sketches with the detail and notation necessary to help them reason [33] or to reinforce what they wish to communicate within the design session [33, 28]. Working with low detail enables sketches to be created quickly and modified easily, providing rapid feedback [6, 28]. Too much structure imposed by a formal notation too soon can create unconscious barriers to change, resulting in a less exploratory and broad search for solutions [34].

Designers refine and evolve sketches. The level of detail designers require grows as designers expand their ideas [27]. Refinement is not uniform across a design: portions may exist at varying levels of maturity [28]. Designers appropriate existing sketches, adding new notational elements to capture decisions as they become more committed [11]. For example, designers appropriate lists, evolving them into class diagrams by first introducing boxes to denote entities and then lines to record relationships between entities (Figure 2). Evolving sketches is unplanned, occurring in response to the needs of the design process [23].

Designers use impromptu notations. Designers work not only with formal notations (e.g., UML), but deliberately break with these to capture ideas in the moment [11]. Beyond annotations and minor deviations, designers sometimes adapt whole notations on the fly, often to describe a problem domain for which there is no standard.

How Designers Navigate Sketches

Designers work with different perspectives. Designers use sketches of varying types to present multiple perspectives on a design, making details hidden in one perspective pronounced and easier to understand in another [28]. For example, in designing a user interface component, designers simultaneously work with views of the user interface and a UML model describing its data model.

Designers work with alternatives. Designers generate sketches of competing alternatives, allowing them to manage their focus, compare alternatives, weigh their tradeoffs, and synthesize alternatives into new alternatives [24, 4].

Designers work with sketches at different levels of abstraction. As designs are often hierarchic, designers work with sketches spanning levels of abstraction, including sketches of user interfaces and architecture [8, 28].

Designers perform mental simulations. Mental simulation provides insight into the consequences of a design, allowing designers to "interrogate" their design by testing it with hypothetical scenarios and inputs, often annotating their sketches [35]. For example, while discussing the logic cars use to move through intersections, a designer may simulate the car's path by moving his finger along a path through a map while simultaneously enumerating the logic required to implement this behavior. Mental simulations help to discover implicit assumptions and flaws in a design [28].

Designers juxtapose sketches. Designers often juxtapose sketches spanning perspectives, alternatives, and abstractions to reason about how a design might work, using information from one to identify inconsistences, omissions, and mistakes in others [28]. For example, designers may use a data model and map to understand how a car object is passed between entities as it travels through an intersection.

Designers review their progress. During a design session, designers sometimes pause to take a step back and consider the progress they have made and what they have yet to do [23]. For example, they may return to requirements lists, marking off those they have been addressed, enumerating those yet to be addressed, and adding additional items.

Designers retreat to previous ideas. When designers become stuck or exhaust an alternative, designers may choose to return to a previous state of the design (and its sketches) [35]. Returning to past designs may bring new insight and a matured understanding to explore the past ideas further.

How Designers Collaborate with Sketches

Designers switch between synchronous and asynchronous work. Design at the whiteboard often occurs synchronously, with designers working together on a single aspect of the design [10]. Designers sometimes break away to asynchronously explore an idea by themselves [14].

Designers bring their work together. After working asynchronously, designers may need to integrate separate ideas into a new unified design. This may involve simply combining parts of several sketches or generating a new design that borrows conceptual aspects. [11].

Designers explain their sketches to others. When returning from independent work or when drawing on behalf of a group, designers must synchronize their mental models of the design by explaining their work to others [11]. Explanations are often supplemented by pointing or drawing on sketches, guiding attention to specific parts of a sketch.

CALICO

Designers use physical whiteboards for their fluidity and flexibility. Our key goals in designing Calico were to maintain this fluidity and flexibility – allowing designers to focus on the content of their sketch rather than the tool used to make it – while enabling users to discover interactive features that help them to design more effectively.

Building on experiences with a previous version of Calico [23], this paper presents a new system redesigned and reimplemented from scratch to support not 4 but 14 distinct design behaviors. To make manipulating content more fluid, we introduce selection scraps and posthoc scrap creation, make scrap interactions more discoverable through bubble menus, and introduce text and list scraps. To support more effectively working with and navigating between perspectives, alternatives, and abstractions while performing mental simulations, juxtaposing, reviewing progress, and retreating to past ideas, we introduce the cluster view. To support more effectively collaborating with sketches, we enable synchronous and asynchronous collaboration across multiple devices and introduce the fading highlighter to help designers explain sketches. In the following sections we describe the features of Calico in detail.

Sketching

As in a physical whiteboard, the most prominent feature of Calico is an open canvas, allowing designers to immediately create a stroke simply by dragging their pen. Designers can select pen color, stroke width, and pen modes and may erase strokes, undo, and redo.

A central benefit of an interactive whiteboard is the interactivity it affords – the ability to move, copy, rotate, and resize. Drawing tools often enable this through modes, allowing users to toggle between drawing and selection modes. However, modes distract from the fluidity a whiteboard provides – designers can no longer stay focused on the design task at hand and must instead maintain awareness of and actively switch between modes.

To minimize this distraction, Calico provides a lightweight selection and manipulation mechanism, allowing designers to select regions of content by circumscription, creating a *selection scrap* (Figure 3b). When a stroke is sufficiently long, a landing zone appears (Figure 3a); ending the stroke inside creates a selection scrap. Calico also enables scraps to be created from existing strokes, either to recover if the user has missed the landing zone or to promote previously created content into a scrap. Pressing-and-holding the pen inside a stroke that circumscribes an area triggers a dotted red circle to appear, which can be tapped to create a scrap. Scraps are inspired by Translucent Patches [20], which allows users to explicitly declare an area as a group. Scraps are movable, copy-able, deletable, rotatable, and resizable, using the bubble menu surrounding the scrap (Figure 3b).

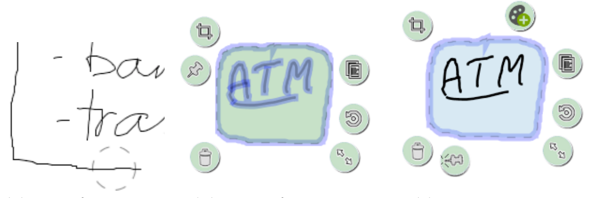
When a selection scrap loses focus, it immediately disappears and returns its content to the canvas, providing interactivity benefits without forcing content to be a persistent object. To permanently retain the scrap, users may tap either of the two scrap icons in the upper left of the bubble menu to transform it into a regular scrap (indicated with a blue background – Figure 3c), either retaining the original shape or creating a neater rectangle.

Once made a regular scrap, a scrap becomes a group that is manipulatable (as described before), stackable, and connectable. For example, the ATM scrap in Figure 3d was first drawn on the canvas, then circumscribed by the stylus to create a regular scrap. Moving a scrap to a position where it is entirely overlapped by another scrap attaches it to the scrap behind it, allowing users to quickly create a *stack* (thereby creating hierarchically composed groups), as one would a pile of papers. Continuing the example, the Deposit, Withdrawal, and CheckBalance scraps are stacked on the Transactions scrap; moving “Transactions” moves the entire stack. Dragging a scrap off a stack ungroups it. For example, moving the scrap labeled “Deposit” from its current location to “User Interface” re-parents it to the new scrap. Scraps do not slide under other scraps; dragging a scrap implicitly moves it to the front.

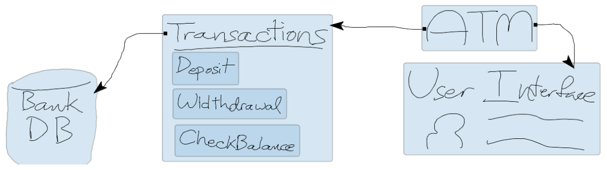
Dragging the pen between scraps highlights the pen stroke, presenting the user with an option to transform the stroke into a connector, through an ignorable button. As with scraps, this can also be done retroactively by press-and-holding a stroke that connects scraps. Connectors preserve the shape of the stroke, but are decorated with an arrowhead. Connectors are persistent and anchored to scraps: moving a scrap resizes the connector.

List scraps enable users to organize a stack into a vertical list, whose boundaries are automatically updated (Figure 4). Promoting a stack into a list organizes the immediate children of the parent scrap into a vertical list. As with the implicit grouping of regular scraps, dragging a scrap onto a list adds it, refreshing the automatic layout. List items also gain an associated box that can be checked and unchecked. Lists can be nested to create multi-level hierarchies.

Text scraps enable users to create typed content quickly from the keyboard, simply by pressing the enter key and



(a) landing zone (b) selection scrap (c) regular scrap



(d) scrap stacks

Figure 3. Scraps allow users to manipulate content.

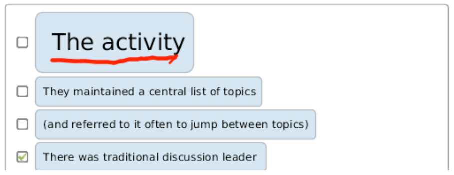
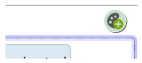

☐ The activity
☐ They maintained a central list of topics
☐ (and referred to it often to jump between topics)
☑ There was traditional discussion leader

Figure 4. List scraps organize scraps into a vertical list.


(a) Palette icon

(b) Palette bar with scraps

Figure 5. Clicking on the palette icon (a) on a scrap’s bubble menu adds it to the palette bar (b).

typing. If a list scrap is selected, the text scrap is appended. Calico also enables scraps to be created from images.

Calico provides a palette, allowing designers to save a scrap for reuse (Figure 5). Dragging a scrap from the palette to the canvas creates a copy of the scrap. The palette is global to all canvases and users, enabling scraps to be shared.

Navigating Sketches

Calico allows designers to create and work with multiple canvases. While working in a canvas, tapping “new canvas” or “copy canvas” navigates to the new canvas and allows sketching to continue. Calico also provides a history stack with buttons to navigate forwards and backwards. Designers may choose to name their canvas with a title.

As designers create many canvases, the set of canvases may become unwieldy. To organize canvases, Calico provides a three level hierarchy: the wall, clusters, and canvases. The wall provides a zoomable, high-level grid view of clusters, allowing designers to move between separate spaces for a project or person (Figure 6). Dragging a canvas between clusters moves it, allowing users to create new clusters and automatically deleting empty clusters.

Tapping a cluster invokes the cluster view (Figure 7) providing a zoomable overview of a group of canvases. Clusters automatically arrange canvases into a radial layout, ordering canvases along concentric circles. In preliminary testing, users reported that clusters provided a meta-design space and wished to organize canvases as part of their design process. Calico thus allows canvases to be manually repositioned, pinning their location.

Calico enables users to construct a narrative describing the relationships between canvases through *tagging*. When a new canvas is created, users are prompted to tag the canvas with its relationship to the previously visited canvas (Figure 8). The tag panel is populated with a set of tags drawn from ways in which designers have been found to relate sketches, including different alternatives, perspectives, and abstractions. The user, however, may add, edit, or delete types of tags. After choosing a tag, the new canvas is linked to the previous canvas in the cluster view, with a label denoting the tag (left side of Figure 7). Repeatedly creating and linking canvases forms a graph structure in the radial layout.

Calico also helps users to find canvases. Navigation history is recorded, and the most recently visited canvas is highlighted with a blue halo in the cluster view (left side of Figure 7). The breadcrumb bar at the top of the canvas and cluster views (Figure 9) let designers directly navigate to any canvas within the hierarchy.

Collaborating with Sketches

Calico supports collaborative work across multiple devices, allowing multiple designers to work synchronously on the same canvas or asynchronously on different canvases. This allows designers working in a group to branch off to their own canvas, preventing designers from "spin[ning] their wheels" while others have the floor [55]. Calico allows user to copy or create a new canvas, work asynchronously, and later invite others to visit the new canvas. Canvases can also be shared by email or by generating a unique URL.

A fading highlighter allows users to draw temporary marks immediately visible to all users currently viewing a canvas. Marks disappear after 4 seconds. This enables designers to annotate sketches during mental simulations, reviews of progress, and explanations, particularly when working in a group with multiple devices or distributed across locations.

Implementation

Calico is implemented as a Java application, spanning approximately 100,000 lines of code and built on the Piccolo UI toolkit for zoomable interfaces [2]. Calico uses a client-server architecture, supporting up to 20 simultaneously active users. The Calico client is portable, supporting computers connected to electronic whiteboards, laptops, and tablets. Calico is open source and freely available[1].

EVALUATION

To evaluate Calico and explore the opportunities and challenges in supporting informal design with interactive whiteboards, we conducted a field deployment of Calico.

[1] https://github.com/uci-sdcl/Calico

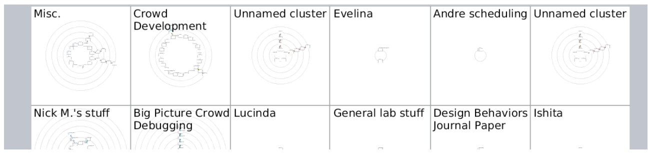


Figure 6. The wall of clusters (partial view).

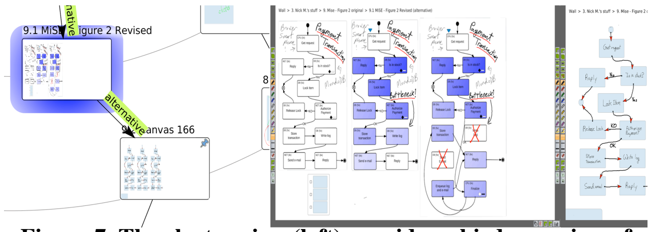
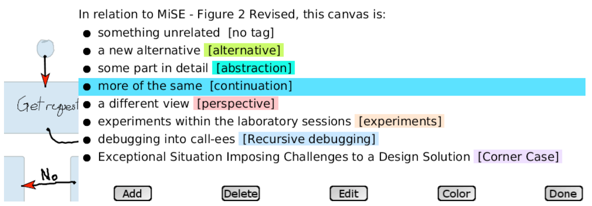


Figure 7. The cluster view (left) provides a birds eye view of two alternative canvases (center and far right).

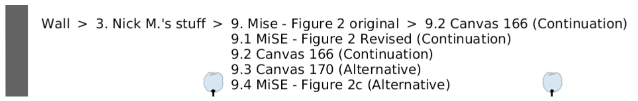


In relation to MiSE - Figure 2 Revised, this canvas is:

- something unrelated [no tag]
- a new alternative [alternative]
- some part in detail [abstraction]
- more of the same [continuation]
- a different view [perspective]
- experiments within the laboratory sessions [experiments]
- debugging into call-ees [Recursive debugging]
- Exceptional Situation Imposing Challenges to a Design Solution [Corner Case]

Add Delete Edit Color Done

Figure 8. The tag panel appears in newly created canvases.



Wall > 3. Nick M.'s stuff > 9. Mise - Figure 2 original > 9.2 Canvas 166 (Continuation)
9.1 MiSE - Figure 2 Revised (Continuation)
9.2 Canvas 166 (Continuation)
9.3 Canvas 170 (Alternative)
9.4 MiSE - Figure 2c (Alternative)

Figure 9. The breadcrumb bar in the upper-left allows users to rapidly navigate between canvases.

Method

We deployed Calico to three groups. In the *research group* (which included an author not associated with Calico at the time), three researchers designing a software development IDE used Calico for over a year, seven months of which was included in the study period. The group was geographically distributed across two sites, but also made extensive use of Calico during a one-week collocated period. In the *interaction group*, two designers at an interaction design firm used Calico over a five-day period. The interaction group used a version of Calico for most of the study period that did not contain the cluster view (including only a two level hierarchy with a grid and canvases); we thus do not report on their use of the cluster view. In the *OSS group*, five software developers at a healthcare open source software company used Calico for a four-week period.

The research, interaction, and OSS groups were setup with two Hitachi Starboard FXDUO77 whiteboards (adjacent in a room), one Hitachi Starboard FXDUO88, and one Hitachi Starboard FX, respectively. Each group also had access to a traditional physical whiteboard, pen-based tablets, and a server instance of Calico. During the study period, we collected usage logs of Calico, recording the complete history of designers' interactions with Calico.

To analyze this data, we first used the logs to probe designers' use of Calico, examining instances both where usage was aligned with the design behaviors and which indicated

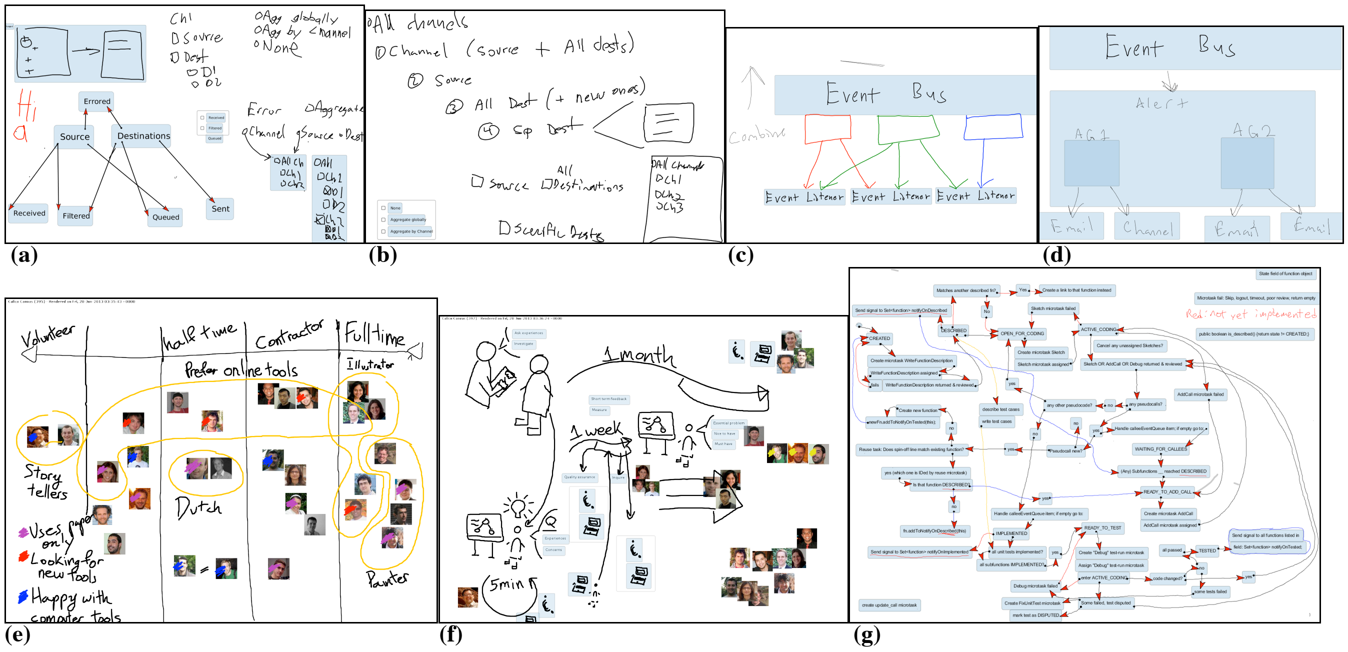


Figure 10. Examples of canvases created by the OSS group (a, b, c, d), interaction group (e, f), and research group (g). To preserve the confidentiality of the canvases, Figures (e) and (f) were recreated, preserving the visual structure but using fictional content.

another intention. After the study period was concluded, we conducted semi-structured interviews with designers in each group, focusing on memorable design experiences with Calico, explanations of interesting behavior observed in the usage logs, obstacles or surprises designers perceived in their use of Calico, how they felt Calico impacted their design process, and perceptions of Calico's features.

Results

Designers made extensive use of Calico (Table 1), with the research, interaction, and OSS groups creating a total of 79, 20, and 40 canvases, respectively. Given the choice between Calico and their traditional physical whiteboards, the interaction designers exclusively used Calico while the research and OSS groups used both, more due to ease-of-access in the moment than due to a preference of use for specific tasks. While designers used Calico over much of the study periods, use was highly concentrated in bursts of activity around meetings, where designers prepared sketches the day before, used Calico intensely during meetings, and reviewed sketches following the meeting. While much of Calico's value came from sketching in the moment, all groups emailed images of canvases to archive their sketches. The interaction and OSS groups did not arrange canvases into separate personal spaces; the research group, which used Calico over the longest period, did. In the following sections, we examine Calico's effect on each of the design behaviors, challenges designers experienced using Calico, and designers' overall impression of Calico.

Sketching

Designers draw different kinds of diagrams. Designers used Calico to create a wide variety of sketches – box and arrow diagrams, UI mockups, lists, tables, use case diagrams, source code, plots, dendrograms, flowcharts, storyboards, and timelines (Figures 1, 10, 11). Designers made use of scraps to organize and arrange content. For example, the interaction group created image scraps of people they had interviewed, organized them along themes, and drew on the diagrams to capture these ideas (Figure 10(e, f)). In the OSS group, designers used scraps to create box-and-arrow diagrams and user interface mockups while brainstorming the elements and appearance of a GUI (Figure 10a). They reported that depicting elements as scraps made them easier to move and resize, making them feel more like entities.

Designers draw what they need, and no more. Inter-

Feature	Total use		Days used		Median use per used day		Max use per used day	
	OSS group	Rsrch group	OSS group	Rsrch group	OSS group	Rsrch group	OSS group	Rsrch group
Strokes	6256	23915	45%	38%	78.5	146	2739	1893
Scraps	1636	14178	29%	31%	41	93	519	2076
Palette	41	360	6%	10%	20.5	3.5	26	114
Fading highlighter	513	212	19%	5%	32.5	11	299	105
Cluster view	1374	10067	42%	53%	26	27.5	626	1470
Overall	10093	60892	45%	54%	137.5	160.5	4212	4701

Table 1. Use of Calico by the OSS and research groups (data for the interaction group is not available; data for the research group includes the first 6.5 months of the 7 month study period). Each unit of activity corresponds to a user action (e.g., drawing a stroke, resizing a scrap, switching between canvases). The overall row includes all interactions with Calico (including activity types not listed).

viewed about their sketches, there was often a large disparity between designers' mental models of what they designed and what the sketches explicitly captured. While designers from the OSS and research groups had difficulty identifying the meaning of some sketches, they recalled the overall objective, which they considered more important than the details. These sketches were used to support activity while "in the moment". For example, the OSS group expressed most of their software architectures using only boxes and arrows (Figure 10(a, c, d)), only rarely labeling the connecting arrows. Most design occurred verbally, and designers only added the detail required to have something to point at during discussion. The OSS group made extensive use of the fading highlighter, permitting discussing and tracing paths over diagrams while preserving their low detail.

Designers varied in the level of detail they used. When drawing similar sketches, designers used inconsistent levels of detail. For example, the interaction group sometimes labeled the axes of plots in detail and other times in very little detail. In other situations, designers created elaborate sketches that visually encoded a wide range of information. A participant in the research group reported that scraps and connectors led them to create more complex sketches, helping them address a deeper level of complexity.

Designers refine and evolve sketches. Designers sometimes began sketches simply, evolving them over time to more complex sketches. For example, the OSS group first created a sketch containing only handwritten names. It then evolved, as the sketched names became text scraps and connectors were added (Figure 10a). The interaction designers often began with pictures of faces, which they then categorized using visual structures. In one example (Figure 10e), they began with a single dimensional line, added categories to the line, and transformed it into a table. While they did not set out to create a table, their design process ultimately led them to create it. Scraps played an important role in this process, helping designers to organize and manipulate content as it evolved. However, designers did not make all content into regular scraps. Designers rarely made complex, handwritten structures such as plots regular scraps, as scraps were a poor fit for these structures.

Designers use impromptu notations. All groups created visual languages in their designs, encoding their own meaning into notations. For example, designers circled scraps (Figure 10e); used color coded lines, underlines (Figure 10g), and boxes (Figure 10c); and dashed lines (Figure 11). The meaning of the notations was often not obvious and sometimes forgotten. A designer in the OSS group reported that he could not recall the meaning afterwards, but felt that it had supported his thinking during design.

Designers sometimes used the palette to record notations that could not be quickly sketched. For example, the interaction designers saved and reused images of people, and the OSS group identified and reused "important entities".

Navigating Sketches

Designers work with different perspectives. All groups shifted their focus among multiple canvases representing different perspectives on their design. For example, the interaction designers shifted focus between canvases categorizing their data using different visual structures (e.g., tables, one and two dimensional plots; Figure 10(e, f)). All three groups found copying canvases useful, enabling, for example, the interaction designers to use a template canvas to rapidly create new canvases to explore new perspectives on their data. The OSS group made frequent use of the cluster view to move between perspectives. When working with canvases, they created chains, providing an order that helped convey a story. This sometimes directly reflected the chronology of their exploration in the design space, while in other cases, designers inserted canvases when they returned to previous sketches and deviated to a new idea.

Designers work with alternatives. Designers in the OSS and research groups used multiple canvases to explore multiple alternatives. In the OSS group, the alternatives were often generated as a result of conflicting opinions during discussion, inspiring a designer to copy a canvas and generate their own interpretation. In contrast, the interaction designers did not use separate canvases to explore alternatives. Unlike the other groups, the alternatives they considered were of different organizations and interpretations of data, which led them to negotiate alternatives verbally rather than through sketching. Finally, one designer reported that not being limited to a single space on a physical whiteboard meant that "*more random ideas get thrown on there,*" increasing the number of alternatives they sketched.

Designers used Calico's tagging feature to label canvases as alternatives. A designer felt that maintaining past alternatives, even when ultimately rejected, was beneficial in providing a record of their design process.

Designers work with sketches at different levels of abstraction. Designers used Calico to work with sketches at varying levels of abstraction, moving both to more and less abstract canvases. For example, the OSS group dove into the behavior of components, copying canvases, and creating new canvases at a lower abstraction level. Designers first started with a more abstract sketch of an event bus connected to event listeners (Figure 10c) before considering the design of a specific "alert" event listener (Figure 10d). All groups used lists – either handwritten or list scraps – to summarize the contents of other canvases, which they re-

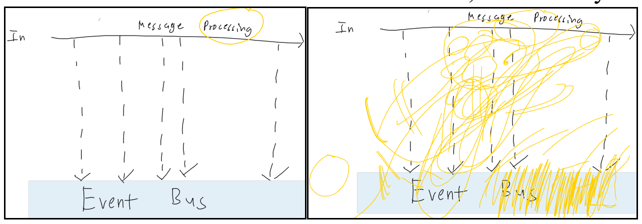


Figure 11. An example of fading highlighter use by the OSS group (left) and a composite of 10 min (of 30) of use (right).

ferred back to while designing.

Designers perform mental simulations. All groups reported that they mentally stepped through their sketches, both verbally in groups and on their own. To do so, the OSS group made heavy use of the fading highlighter. Displaying architectural sketches on the large electronic whiteboard during a meeting, they discussed a sketch at length, gesturing at components with their hands and using the fading highlighter from a tablet that was remotely connected to the same canvas. In one instance, they discussed a single sketch for 30 minutes using the highlighter (Figure 11).

Designers juxtapose sketches. All groups juxtaposed sketches, either navigating back and forth between sketches or copying dispersed content onto a single canvas using the palette. For example, a designer in the research group copied pieces of a process flow and used an adjacent table to step through the diagram (Figure 1). In some cases, juxtaposed sketches served as a static reference in creating a new sketch; in other cases, designers evolved both in parallel.

Designers review their progress. All groups reported that they reviewed their progress. Most used lists (most often handwritten or as text scraps) to summarize aspects of their design, which they sometimes referenced and updated. Designers also reviewed their progress by rapidly moving back and forth between several canvases or by using the cluster view for an overview. While not sufficiently detailed to examine canvas content, the cluster view anchored discussion and allow designers to gesture at canvases, with the linkages between canvases helping designers to recall "how the session played-out".

Designers retreat to previous ideas. Only designers in the multi-week, long-term design sessions (the OSS and research groups) retreated to previous ideas, reporting that they did not return to previous ideas until a later design session, at which point Calico helped to refresh their memory of their past approaches. Both reported that, since they did not feel a need to delete unused sketches, they returned to old sketches more often. The graph structure provided by the cluster view helped designers to locate old sessions and remember their meaning, with linked canvases assisting in reconstructing meaning. A designer in the research group reported:

"Designs get very complex... you want to keep a history of what you've done, the branches that you've pruned… If you're designing a complex thing with stages and you're trying to tell a story, you can say: okay we've tried that.. If you don't have the structure you'll have to create it somewhere else. [You save time] if it's already here..."

Collaborating with Sketches

Designers switch between synchronous and asynchronous work. Designers in the research and OSS groups used Calico across multiple devices. For the OSS group, this led to a more informal setting in which members spontaneously broke into small groups in meetings, handing tablets back-and-forth, sketching over the diagrams, and displaying their annotations on the electronic whiteboard. With multiple tablets, multiple team members could talk simultaneously without a single arbiter at the whiteboard blocking content production, an issue in whiteboard use [30]. In the distributed research group, this enabled remote participants to be more active by sketching ideas. In contrast, the interaction designers were collocated and had an established culture of working in pairs, leading them to not break into groups.

The OSS group reported working asynchronously at least once every session and felt that it was an important benefit:

"The fact that someone can work with their own tablet or computer… is something really powerful… Especially when someone is already at the whiteboard discussing something and you want to bring in an alternative perspective but you need to wait until they're done."

Designers bring their work together. Designers rarely did this, as the interaction group did not work asynchronously and the research group did not combine their work. However, the OSS group twice combined work produced asynchronously, creating a new canvas, linking it to the previous canvases with tags, and summarizing their work.

Designers explain their sketches to others. All groups explained their sketches to one another but varied in the situations in which they did so. The interaction designers worked exclusively synchronously, explaining designs only when a designer challenged decisions. The OSS group sometimes worked asynchronously and used explanations when returning to synchronous work. The research group worked more independently and explained days of work to other team members. In most cases, designers explained their sketches by pointing, gesturing in the air, or simply verbally, with the fading highlighter sometimes assisting.

Challenges using Calico

Our study revealed a number of weaknesses in Calico, ranging from usability issues to challenges inherent to interactive whiteboards. The interaction designers reported that rapidly rearranging many scraps was not well supported, as the gesture of moving scraps (click and hold) could be slow. Due to the cluster view's layout approach, it often zoomed out far to show all canvases, making it difficult or impossible to read the content on individual canvases. This made juxtaposing sketches more challenging, forcing designers to explicitly copy canvases using the palette or to rapidly jump between canvases. It also made simply navigating between canvases using the cluster view more challenging. Designers also wanted the ability to more easily augment the set of tags, to for example, declare which alternative was chosen.

While the fading highlighter played an important role in several situations, designers often felt that they forgot to use it "in the heat of the moment". Moreover, it was sometimes

Empirical Result	Design Recommendation
Designers simulate and discuss scenarios very frequently.	Enable annotating sketches with multiple scenarios.
Interactive whiteboards diminish handwriting quality.	Enable alternative text input (e.g., speech to text or text recognition)
Designers work simultaneously with several canvases.	Enable multiple canvases to be legibly viewed simultaneously.
While separating sketches across canvases has important benefits, multiple canvases are sometimes parts of a single sketch.	Enable designers to expand canvases when necessary.
Designers use impromptu notations whose meaning is forgotten when sketches are reviewed.	Enable designers to reconstruct meaning by recording and replaying audio from design sessions.
Determining the authorship of content is challenging.	Provide authorship cues as content is created.
Designers work synchronously and asynchronously, moving together between canvases and working on separate canvases.	Enable designers to temporarily subscribe to a group focus.

Table 2. Empirical results on informal design with interactive whiteboards and recommendations for design.

confusing which designer was drawing – designers wished to see a name associated with highlights. While the cluster view depicted the current canvas of each device, the designers still felt slowed down when moving between canvases with multiple participants, requiring that they announce what canvas they were moving to.

Nearly all groups reported that the large electronic whiteboards diminished the quality of their handwriting, forcing them to write slower or larger, write with a tablet, or enter text using a keyboard. The interaction designers found the space available too small, reporting that they were *"blocked by the physical limitations of the [electronic] board."*

Overall Impressions

The research group and OSS group both felt that, on balance, the benefits of using Calico outweighed its difficulties and wished to continue to use Calico in the future. The research group felt that Calico helped support their meetings. Prior to using Calico, the group used physical whiteboards and emailed picture of the whiteboard to the remotely located team member. They preferred Calico over a formal diagramming tool as they wished to maintain informality and the ability to freely sketch. The OSS group reported that they did not feel any loss of expressive control in using Calico in comparison to the whiteboard, and reported that they normally would have performed many of the same activities on physical whiteboards in their meeting spaces.

The interaction designers reported that they would not continue to use Calico, as it did not match their needs. They wished to have infinitely sized canvases – which Calico did not provide – and felt trapped by the limited space. Further, performance was slowed by using a large number of images on a single Canvas, making Calico less responsive.

DISCUSSION

Through a review of the software design literature, we identified 14 behaviors that characterize informal design at the whiteboard and designed an interactive whiteboard system – Calico – to support these behaviors. Through a deployment of Calico to three groups of designers, we examined how supporting these behaviors impacts the practice of informal design. We found that, by supporting these behaviors, interactive whiteboards can help designers to more effectively manipulate content, work with groups and relationships amongst sketches, and collaboratively design synchronously and asynchronously.

Our field deployment revealed several challenges in supporting informal design, suggesting several design recommendations beyond supporting the design behaviors (Table 2). For example, designers constantly use general-purpose sketches to simulate and discuss scenarios, annotating and tracing paths over sketches. This might be more effectively supported by allowing designers to use and reference multiple scenarios on top of general-purpose sketches. As another example, diminished handwriting quality remains an important issue, suggesting the need to consider alternative mechanisms for text entry such as speech to text. Together, the design behaviors and design recommendations provide guidance on how informal design can be effectively supported with interactive whiteboards.

ACKNOWLEDGEMENTS

We thank all of the designers who participated in our study. This research was funded in part by the National Science Foundation under grants CCF-1118052 and IIS-1111446.

REFERENCES

1. Baker, A., van der Hoek, A., Ossher, H., and Petre, M. Guest editors' introduction: Studying professional software design. *IEEE Software 29*, 1 (2012), 28–33.
2. Bederson, B., Grosjean, J., and Meyer, J. Toolkit design for interactive structured graphics. *Transactions on Software Engineering 30*, 8 (2004), 535–546.
3. Bellamy, R., Desmond, M., Martino, J., Matchen, P., Ossher, H., Richards, J., and Swart, C. Sketching tools for ideation. In *Proc. ICSE 2011 (nier track),* 808–811.
4. Buxton, B. Sketching User Experiences: Getting the Design Right and the Right Design. Morgan Kaufmann, 2010.
5. Chen, Q., Grundy, J., and Hosking, J. Sumlow: early design-stage sketching of uml diagrams on an e-whiteboard. *Software: Practice and Experience 38*, 9 (2008), 961–994.

6. Cherubini, M., Venolia, G., DeLine, R. and Ko, A. J. Let's go to the whiteboard: how and why software developers use drawings. In *Proc. CHI 2007*, 557–566.

7. Chung, R., Mirica, P., and Plimmer, B. Inkkit: a generic design tool for the tablet pc. *In Proc. New Zealand's Conf. on CHI 2005*, 29–30.

8. Da Silva, P. P. User interface declarative models and development environments: A survey. In *Proc. Int. Systems Design, Specification, and Verification 2001*, 207–226.

9. Damm, C. H., Hansen, K. M., Thomsen, M., and Tyrsted, M. Supporting several levels of restriction in the uml. In *Proc. UML 2000*, 396–409.

10. Dekel, U. Supporting distributed software design meetings: what can we learn from co-located meetings? In *Proc. HSSE 2005,* 1–7.

11. Dekel, U., and Herbsleb, J. D. Notation and representation in collaborative object- oriented design: an observational study. In *Proc. OOPSLA 2007*, 261–280.

12. Geyer F., Budzinski, J., and Reiterer, H. Ideavis: a hybrid workspace and interactive visualization for paper-based collaborative sketching sessions. In *Proc. Nordic CHI 2012,* 331–340.

13. Guimbretière, F., Stone, M., and Winograd, T. Fluid interaction with high-resolution wall-size displays. In *Proc. UIST 2001*, 21–30.

14. Gumienny, R., Gericke, L., Wenzel, M., and Meinel, C. Supporting creative collaboration in globally distributed companies. In *Proc. CSCW 2013,* 995–1007.

15. Hailpern, J., Hinterbichler, E., Leppert, C., Cook, D., and Bailey, B. P. Team storm: demonstrating an interaction model for working with multiple ideas during creative group work. In *Proc. Conf. on Creativity and Cognition 2007*, 193–202.

16. Hammond, T., and Davis, R. Ladder: A language to describe drawing, display, and editing in sketch recognition. In *Proc. ICJAI 2003,* 461-467.

17. Huang, E. M., Mynatt, E. D., Russell, D. M., and Sue, A. E. Secrets to Success and Fatal Flaws: The Design of Large-Display Groupware. *IEEE Computer Graphics and Applications 26*, 1 (2006), 10–17.

18. Johnson, G., Gross, M. D., Hong, J., and Do, E. Y.-L. Computational support for sketching in design: a review. *Foundations and Trends in Human Computer Interaction 2*, 1 (2008), 1–93.

19. Klemmer, S. R., Newman, M. W., Farrell, R., Bilezikjian, M., and Landay, J. A. The designers' outpost: a tangible interface for collaborative web site. In *Proc. UIST 2001*, 1–10.

20. Kramer, A. Translucent patches—dissolving windows. In *Proc. UIST 1994*, 121–130.

21. Landay, J. A., and Myers, B. A. Interactive sketching for the early stages of user interface design. In *Proc. CHI 1995,* 43–50.

22. Loksa, D., Mangano, N., LaToza, T. D., and van der Hoek, A. Enabling a classroom design studio with a collaborative sketch design tool. In *Proc. ICSE 2013*, 1073-1082.

23. Mangano, N. and van der Hoek, A. The design and evaluation of a tool to support software designers at the whiteboard. *Automated Software Engineering 19*, 4 (2012), 381–421.

24. Myers, B., Park, S. Y., Nakano, Y., Mueller, G., and Ko, A. J. How designers design and program interactive behaviors. In *Proc. VL/HCC 2008,* 177–184.

25. Mynatt, E. D., Igarashi, T., Edwards, W. K., and LaMarca, A. Flatland: new dimensions in office whiteboards. In *Proc. CHI 1999,* 346–353.

26. Newman, M. W., Lin, J., Hong, J. I., and Landay, J. A. Denim: an informal web site design tool inspired by observations of practice. *Human Computer Interaction 18,* 3 (2003), 259– 324.

27. Ossher, H., John, B., Desmond, M., and Bellamy, R. Are flexible modeling tools applicable to software design discussions? In *Workshop on Studying Professional Software Design 2010*, volume 12.

28. Petre, M. Insights from expert software design practice. In *Proc. ESEC / FSE 2009*, 233–242.

29. Sangiorgi, U. and Vanderdonckt, J. Gambit: Addressing multi-platform collaborative sketching with html5. In *Proc. EICS 2012,* 257–262.

30. Shih, P. C., Nguyen, D. H., Hirano, S. H., Redmiles, D. F., and Hayes, G. R. Groupmind: supporting idea generation through a collaborative mind-mapping tool. In *Proc. CSCW 2009*, 139–148.

31. Stefik, M., Foster, G., Bobrow, D. G., Kahn, K., Lanning, S., and Suchman, L. Beyond the chalkboard: computer support for collaboration and problem solving in meetings. *CACM 30*, 1 (1987), 32–47.

32. Streitz, N. A., Geissler, J., Haake, J. M., and Hol, J. Dolphin: integrated meeting support across local and remote desktop environments and liveboards. In *Proc. CSCW 1994,* 345–358.

33. Tversky, B. What do sketches say about thinking. In *Proc. AAAI Spring Symposium 2002, Sketch Understanding Workshop*, 148–151.

34. Wong, Y. Y. Rough and ready prototypes: Lessons from graphic design. In *Proc. CHI 1992*, 83–84.

35. Zannier, C., Chiasson, M., and Maurer, F. A model of design decision making based on empirical results of interviews with software designers. *Information and Software Technology 49*, 6 (2007), 637–653.