An Activity Centered Approach to Nonvisual Computer Interaction

MARK S. BALDWIN, University of California
JENNIFER MANKOFF, University of Washington
BONNIE NARDI and GILLIAN HAYES, University of California

In this work, we apply an activity theory lens to analyze nonvisual computing for blind and low-vision computer users. Our analysis indicates major challenges for users in translating the activities they are working towards into specific tasks to be completed in a system comprehensible manner. Specifically, blind and low-vision students learning to use accessible technologies struggled with organizing their activities, tracking the history and status of their operations, and understanding how the system was acting underneath these interactions. We discuss how activity-centered design can be applied to nonvisual interfaces to better match user behavior in a computational system.

CCS Concepts: • **Human-centered computing** → **Haptic devices**; **Empirical studies in accessibility**; **Accessibility systems and tools**; *HCI design and evaluation methods*; • **Social and professional topics** → **People with disabilities**;

Additional Key Words and Phrases: Accessibility, visual impairment, blindness, activity theory, activity based computing, assistive technology

ACM Reference format:
Mark S. Baldwin, Jennifer Mankoff, Bonnie Nardi, and Gillian Hayes. 2020. An Activity Centered Approach to Nonvisual Computer Interaction. *ACM Trans. Comput.-Hum. Interact.* 27, 2, Article 12 (March 2020), 27 pages.
https://doi.org/10.1145/3374211

1 INTRODUCTION

From the graphical user interface and the world wide web to smartphones and voice assistants, ensuring the accessibility of computational systems is in constant pursuit of technological advancement. Despite continued improvements to screen readers, screen magnifiers, and other technologies, nonvisual computing for blind users and those with limited vision continues to lag behind visual computing in usability and advanced functionality [11]. One explanation for these challenges lies in the fundamental structures of computation, traditionally hierarchical, and now increasingly oriented towards search and sensor streams. Rather than accept these technological orientations, a computational system structured around *the unit of human activity* [50] holds promise as a way to remove many of the obstacles blind and low vision users face.

This work was supported by the Kleist endowment and the Jacobs Foundation Advanced Research Fellowship.
Authors' addresses: M. S. Baldwin, B. Nardi, and G. Hayes, University of California, Irvine, 5019 Donald Bren Hall, CA, 92697; J. Mankoff, University of Washington, Paul G. Allen Center, 185 Stevens Way, Campus Box 352350, Seattle, WA 98195.

1073-0516/2020/03-ART12 $15.00
https://doi.org/10.1145/3374211

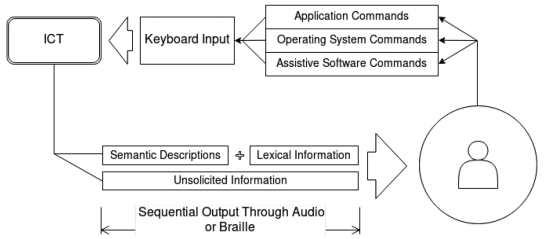


Fig. 1. The typical workflow for nonvisual use of computational systems. A blind or low vision user must recall application, operating system, and assistive software keyboard commands to control input into an Information and Communications Technology (ICT). The ICT responds to keyboard input through a serial stream of auditory information that combines semantic and lexical information.

The mismatch between activity and computational infrastructures for blind and low vision computer users can be extreme. While visual computing users take advantage of large screens, multiple monitors, and multiple desktops to orient themselves visually to swap quickly among a variety of contexts and activities, blind and low vision users must rely on a keyboard for input and specialized software for converting visual and textual information to speech or a magnified viewport (see Figure 1). This approach widens the gap for blind and low-vision users between the human activity and the technological abstraction, as all of the services and data typically communicated visually (*e.g.,* progress bars, notifications, and spatial information), must also be put into the audio channel. With each technological advancement in visual communication, new efforts are required to translate information into a nonvisual medium.

Thus, in this work, we were focused on how we might reconsider accessible technologies for the desktop computing environment from the ground up as an infrastructure centered around activity. In this way, we can narrow the gaps for blind and low-vision users between the goals they wish to accomplish, the tasks they are attempting to complete, and the computing infrastructure they must use. We see two distinct advantages to this approach beyond the obvious improvement of solutions for blind and low-vision users. First, by simplifying abstractions to make them accessible, we may reduce the kind of overhead that is limiting adoption of an activity-centered approach in the broader community. Second, by re-articulating blind and low-vision user needs and preferences through the lens of activity theory, we contribute a conceptual framework for thinking about accessibility and assistive technologies.

In what follows, we first describe the related work in activity theory as well as visually impaired computer use more broadly. We then provide a brief review of our data collection and analysis from ongoing field work with visually impaired computer users. The results of this field work are presented through an activity theory lens to identify key areas in which activity can improve the computing experience. Finally we offer a set of design principles that designers should consider when building tools for nonvisual interaction.

2 RELATED WORK

Viewing desktop accessibility from the perspective of human activity as we present here is a new paradigm; however, this work builds on substantial efforts to improve the computing experience for the visually impaired community within human– computer interaction (HCI). The transition from command line to graphical interface signaled a turning point for individuals who were unable to manipulate a WIMP (windows, icons, menus, and pointers) environment. This change has been the basis for nearly three decades of research in accessibility and assistive technology focused on closing the operational gap between sighted and visually impaired computer users. Such efforts

are particularly noteworthy as economic independence, which is increasingly dependent on basic computer skills, is a critical component of well-being for the visually impaired [16, 21].

There are substantial obstacles towards reaching employment for those with visual impairment. In a survey of over one thousand blind and visually impaired adults, researchers found that less than 40% of surveyed participants were employed [7]. Employment has been shown to be more likely for individuals who had obtained higher levels of education, were fluent braille readers, and had acquired skills with assistive technology [7, 47]. One likely reason for these results could be related to the assistive tools that are available. As most are prohibitively expensive, they are commonly subsidized by state and local agencies, contributing to an already high cost impact [24]. For example, a mid-level braille display capable of functioning with the latest computer systems often costs more than top-end computer systems [31].

2.1 Accessible Systems

Sonification is the use of non-speech audio to convey information or perceptualize data.

Early efforts in accessibility research focused on the development of sub-systems that understood how to pair metadata with interface elements [22, 49]. As sub-systems became natural components of user interface toolkits,[1,2,3] attention turned to making sure that the metadata was adequately communicated. The ubiquity of 16 bit computer audio, for example, saw significant research efforts in the auditory presentation of data including audio manipulation techniques like sonification and 3D audio [9, 19, 20, 25, 68]. The positive results of many of these research efforts is likely due to the capabilities of the human auditory system. Blind people are known to have significantly stronger verbal auditory encoding abilities [57] as well as a stronger ability to identify individual sounds from concurrent speech [28] when compared to sighted individuals. Yet, few of these systems have gained traction in commercial audio-based interaction tools like screen readers. The serial, ephemeral nature of audio does not adequately translate the metaphor and expressiveness of the graphical user interface (GUI).

A tangible user interface (TUI) is a user interface in which a person interacts with digital information through the physical environment.

Tangible computing is one alternative that can lead towards enabling permanence in a nonvisual interface. The human kinesthetic channel is a rich modality capable of interpreting a broad set of sensations [37]. Interacting with physical objects is often found to be an enjoyable experience [60, 72] that strengthens human recall ability [39]. The promise of tangible computer interfaces [72], often exemplified through the works of Ishii [32] and Weiser [69], continues to be explored through a vast array of techniques and technologies [6, 23, 29, 33, 46, 58, 61, 70, 71].

The combination of modalities can offer a richer interaction experience for nonvisual computer users. Auditory and kinesthetic channels combined could create a stronger experience [44]. However, the vast information set presented through the traditional application-document metaphor remains difficult to translate. These complexities are well known outside the assistive technology field of research, leading to many efforts to organize information in more convenient ways [30, 59].

2.2 Activity Theory

Activity theory describes a conceptual framework for understanding the goals, motives, and needs of human consciousness. Activity theorists commonly describe activity as a relationship between the human *subject* and an *object*, traditionally denoted symbolically as $S \rightarrow O$. The needs of the subject motivate actions towards an object, influenced by the attributes of both [67]. An object represents some thing, material or conceptual, that requires change. The actions that humans carry

[1] https://docs.microsoft.com/en-us/windows/uwp/accessibility/accessibility-overview.
[2] http://docs.oracle.com/javase/7/docs/technotes/guides/access/.
[3] https://developer.apple.com/accessibility/macos/.

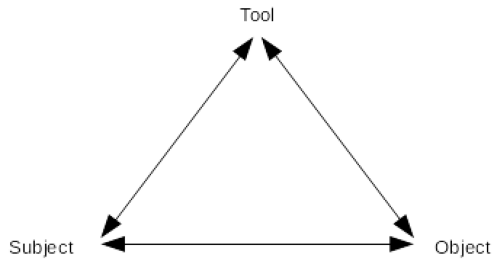


Fig. 2. A diagram depicting the classical representing of the relationship between the subject, object, and mediating tool in activity theory.

out upon the object are predominately driven by mediating artifacts [42] and motivated by goals. According to activity theory, the role of the mediating artifact or tool is to support the subject as it changes the object. The classic paradigm used throughout activity theory literature is the carpenter (subject) who uses a hammer (mediation) to hit a nail (object) [34, 50]. The nail is transformed through the action of being hit which is carried out in service of some goal or goals required to complete the activity; for example, to practice hammering or repair a fence. The relationship between subject, object, and mediation is typically represented by the model in Figure 2.

The application of the principles of activity theory to general computing was a critical component to the post-cognitivist movement within HCI [34]. As computers became part of our everyday lives, so did their role as mediating tools. In her seminal work, Suzanne Bødker outlined the role of the computer as mediating artifact; a user (subject) types words into a computer (mediation) to send an email (object) [10]. The computer as mediator paradigm has formed the basis for a significant body of literature focused on bridging gaps between human activity and computation [17]. Of particular interest to our work are the efforts focused on re-framing desktop interaction from an application-centric model to one centered on activity [4, 36, 65]. Despite promising results, the application-centric model remains largely a siloed experience, where applications and the computational entities that they generate have little awareness of each other, leaving the user to develop individual strategies for coordination [34]. One exception can be found in applications that encapsulate individual tools into a common environment. Personal information managers (PIM) such as Microsoft Outlook, are one example of a single application that can operates at the level of activity [34]. PIMs typically coordinate messaging, scheduling, and social contacts seamlessly together around a single goal, a fundamental requirement of activity. Although the PIM presents a compelling example for the integration of activity principles in computing, there is little evidence of activity centered systems being taken up in a broad and comprehensive sense.

When considering the challenges faced by users of assistive technologies [40, 41, 45], it is clear more work is needed to improve their computing experience. As we conducted our review of the literature, efforts to align assistive technology with activity theory were notably absent. The remainder of this work will demonstrate how activity theory can be used to inform the design of assistive systems.

3 METHODS

This work is part of a series of ongoing extensive participatory projects, working with and for blind and low-vision individuals from a work training program that teaches nonvisual computing skills and a blind school for children staffed with blind adults. In this work, we consider our data in light of the mismatch between activities and the types of tasks nonvisual computing users are required to perform and the potential benefits of an activity-centric approach. In this section, we

Table 1. Students Observed for Field Study

ID	Gender	Age	Technology	Condition	Time with Condition
B1	Female	39	Screen Reader	Glaucoma	9 years
B2	Male	35	Screen Reader	Optic Neuropathy	10 years
B3	Female	43	Screen Reader	Retina Pigmentosa	since childhood
B4	Female	36	Screen Reader	Retina Pigmentosa	since birth
B5	Female	41	Screen Reader	Retina Pigmentosa	7 years
B6	Male	37	Screen Reader	Optic Neuropathy	4 years
B7	Female	41	Screen Reader	Retina Pigmentosa	since birth
M5	Female	34	Magnification	Glaucoma	8 years
B9	Female	45	Screen Reader	Glaucoma	4 years
I1	Male	52	Screen Reader	Unknown	since birth
M3	Male	35	Magnification	Cataracts	since birth

Detailed list of the visually impaired students and instructor observed at the field site. Technology represents the primary tool students preferred to use. Participant ID labels are coded according to technology; B group had less than three months of experience with their assigned technology, M group used magnification, and the I participant was the blind instructor.

briefly describe the empirical data collected as part of this large ongoing project and then detail the specific analysis we conducted for the work presented here.

3.1 Field Study at Empowertech

Our engagement began with a 4-month field study in a blind and low-vision job skills training computer class taught at the Los Angeles based non-profit Empowertech.[4] All of the students in the class were new to screen reading and related technologies they were learning in the course of the class (see Table 1). Over the 4-month period of this study, the first author participated in 48 hours of participant observation across 12 class sessions by working as a teaching assistant for classes of 8 to 12 students each week. The class progressed from learning how to use a screen reader to web navigation, email, and text editing. The first author conducted class-wide observation except when individual students needed assistance, and he would be asked to work with them directly. The tools used within the classroom varied widely between students because all but one student brought a privately owned laptop. Two students used Apple MacBooks; the remaining students used a mixture of Windows 7 systems. Additionally, there were two students who transitioned to new systems: one migrated from Windows 7 to a MacBook, and the other upgraded from Windows 7 to Windows 8. Although the classroom had specialty devices available like a digital magnifier and braille printer, neither were ever used. The primary tools that students relied on were the Victor Reader[5] and screen reader or screen magnifier. The Victor Reader, a pocket-sized handheld digital voice recorder and playback device with built-in text-to-speech, was primarily used by students to record instruction during class. The instructor encouraged students to choose whichever computer, operating system, and assistive tools they preferred.

Following each class, the first author conducted informal interviews with the students about their experiences with technology both in and out of the classroom. Additional artifacts, including handouts and worksheets assigned by the instructor, were collected at each class period and saved for analysis. Between classes, the first author engaged with the most commonly used software

[4]Please see our prior work for a full review of our field study and experiment [3].
[5]http://www.humanware.com/microsite/stream/index.html.

in the previous class to better familiarize himself with the opportunities and challenges the tools provided and to prepare to support students during class.

Field notes were taken during the class when possible and directly after. In-class instruction focused heavily on screen reader operation and keyboard commands for common computing tasks, such as file management, web browsing, and word processing. Outside of class time, the students often stayed at Empowertech to practice their skills, providing additional time for observation of their computer use and for informal interviews. Field notes focused on seating arrangements, lecture topics, software being used, and the interactions between students and the instructor. In our initial analysis, we let the field data structure our findings. The field site structured student activities in highly cognitive, task-based ways, leading to research results and solutions that focused on the actions, inputs, and outputs surrounding specific student goals.

3.2 Experimental Study

In coordination with our field work, we developed a set of tangible devices designed to augment the traditional screen reader with physical representations of visual desktop metaphors [3]. We conducted an experimental study to evaluate the use of our system compared to the participants' personal systems. Each participant was asked to find the lowest price of two preselected products across three different websites. The products were randomly assigned to either our tangible study system or the participant's personal computer. Each participant session was recorded on video, allowing us to capture ways in which our blind and low vision participants configured and operated their personal computers to accomplish essential computing tasks like opening, closing, and switching between applications, windows, tabs. Post study interviews were conducted to elicit feedback about the experience of using our tangible study system. The answers to our interview questions frequently involved discussions about the challenges our participants faced attempting to carry out computational tasks. These discussions revealed significant barriers that the traditional nonvisual computing model places on blind and low-vision users.

3.3 Summary

Given the struggles we observed during our time with Empowertech between student goals and activities and the tasks they were asked to complete, we saw an activity theoretical analysis of our data as essential to a more complete understanding of the challenges and opportunities of nonvisual computing use. Thus, we began this analysis by working to understand the underlying motives [42, 43], needs, and desires of blind and low-vision individuals during our engagements. For example, working in the Empowertech class, we observed a separation between the needs of the students and the stated goals of the class and the specific curriculum being taught. So we focused on the actions taken by students in our fieldwork and experimental study, specifically the processes that they enacted with their goals in mind in service of their overall motives. These actions mapped well to the tasks we had already sought to understand in our previous analysis [3], as is common in HCI research [35]. Finally, we sought to understand the unconscious operations being conducted in the service of these tasks, and where the breakdowns were occurring when considered as a part of the larger whole. Taken together, this analysis indicates (1) the ways in which an activity-centric approach would be particularly supportive of nonvisual computer users, (2) some of the challenges surrounding the shift from application-centric to activity-centric computing that have made it hard to take up more broadly, and (3) multi-modal and tangible solutions that can address the needs of nonvisual computer users specifically but also an activity orientation to computing more broadly.

Place Marker List

Menu (not visible)
Introduction
Discussion (not visible)
More Information

Fig. 3. In the computer display on the left, a typical web page is loaded in a browser and a link object called "More Information" has focus (denoted by the dotted box). The Place Marker List table on the right is a visual representation of user generated place markers that a screen reader stores on the computer. In this depiction, a place marker has just been created for the focused object, by issuing the four key combination: control + shift + MOD + k (MOD is the modifier key assigned by the screen reader). The remaining items in the list represent previously marked objects both on and off the viewable area of the web page. Issuing the key combination MOD + k or MOD + shift + k traverses the place marker list forward and backward, respectively. Notice that place marker order does match overall web page hierarchy.

4 AN ACTIVITY THEORETIC APPROACH TO ACCESSIBILITY

As described in the previous section, using an activity theory lens to examine our fieldwork accentuates the numerous challenges that blind and low-vision computer users experience while attempting to carry out their activities. Through this lens, we pay particular attention to the relationship between actions (the tasks that humans consciously perform) and operations (actions which humans unconsciously complete) [10]. To disambiguate our description of the types of activities we draw from in our data, we apply the term "task" to the actions or operations our participants carried out in service of their goals. The challenges we identify emerge in three primary ways. First, nonvisual computer users often struggle to structure their data and services by activity. The usual techniques of establishing structure through visual arrangement are not available. Second, tracking activities, such that they can be paused and resumed, is challenging without visual markers. Support for tracking, to the extent that it exists in screen reader software, typically requires advanced knowledge of label and marker placement features with functionality varying across applications, screen reader software, and operating systems (see Figure 3). Finally, to understand what the system is doing without visual feedback, users must explicitly query and cycle through the services and data that are available, making the system's behaviors largely opaque. In this section, we describe each of these challenges, as they manifested in our fieldwork and as they relate to the activity theory inspired literature. We then comment on how computational systems might overcome these challenges if built with consideration for activities as the fundamental computational unit and nonvisual access as the core interface.

4.1 Organization by Activity

Humans do substantial work to translate their activities into tasks that can be completed within the hierarchical structure of a computing environment. In many cases, this work involves visual arrangements of materials. For example, when copying materials from one place to another, a sighted computer user might open two windows showing the file structure and place them side by side. Similarly, when comparing prices and selection across online retailers, a sighted shopper is likely to lay out multiple browser instances in tiles or tabs, enabling rapid scanning and comparing

of options by clicking through the visible tabs. In our fieldwork, we observed blind and low vision computer users attempting these same activities but with radically different organizational structures and coping mechanisms. For example, consider the following vignette from our field work:

> The instructor teaches all of the students how to work with a file system, which is largely about learning keyboard shortcuts to "walk" the tree while listening to audio readouts of the folder titles and meta-data. For one assignment, students were asked to create a new folder and save a file to it. The task could be performed either by opening the file browser, creating a folder, and moving the file or using the save file dialog from within the word processing application that was used to create the file. The students encountered numerous challenges as they painstakingly attempted to complete the task. First, locating the appropriate parent directory, creating and naming a new folder, and relocating a file to the desired location must all be completed using keyboard commands to interface with multiple parts of the file system: operating system, file browser, and application. – Field notes

For the sighted user, the flexibility to perform basic file management operations at different contextual levels is advantageous (*e.g.,* creating a folder from within an application, rather than using a file browser.) However, the same flexibility incurs cognitive costs on visually impaired users who must balance command memorization with mental models that do not always match the system state [3]. When mismatches occur, visually impaired users are forced to employ time-consuming reorientation actions to complete their tasks [63]. In the vignette above, students had to learn how to manipulate two conceptually identical mental models using distinctly different interaction patterns. Rather than remember the keyboard commands to save or move a file, they must first cognitively orient to the context (*i.e., file dialog or file browser*), then map the requisite commands that serve the context to complete the action to their mental model. For example, in the Microsoft Windows desktop environment a file dialog does not contain the menu structure present in the file browser. The absence of a menu leads to different focal order when using navigational keys to locate and traverse files and folders. Visually impaired users must negotiate these differences by either memorizing the relevant changes between contexts or through seeking behaviors such as exhaustive scanning [62].

Throughout our field work and experimental study we observed a consistent pattern of file and application organization being carried out on the desktop. The sighted user benefits from a two-dimensional spatial arrangement in which related items can be grouped and clustered into meaningful visual relationships, whereas the nonvisual user relies on the desktop for reorientation and object location. The following observation from our experimental study (Section 3.2) highlights this interaction:

> After opening the first assigned web page the participant said, "Okay, hold on, let me return to the desktop," before receiving the name of the second web page to open. She pressed the windows key followed by the "m" key to return to the desktop, then repeated the key combination she had previously used to open a new instance of Internet Explorer. — Observation from experimental study

Here the participant is setting up her system in preparation for the study. Her approach was to return to the desktop for each new web page using the same three commands. Her desktop reflected the files and programs that she used the most. By memorizing only a few shortcut commands she could quickly reorient to a familiar place—a tactic frequently employed by blind computer users [62]. She could use arrow keys to navigate vertically and horizontally to the desired location. As a novice user, this behavior was likely a result of the training she had received up to that point,

yet it also reveals a simplicity that students learning the file system did not have. The desktop, therefore, served as her activity, the files and application shortcuts located within represented the tools she needed to complete her tasks.

Unfortunately, because traditional desktop systems were not built to treat these behaviors as a single activity, but rather as a series of individual tasks, the responsibility of negotiating the activity hierarchy is placed on the user. In the previous vignette, the participant mixes command memorization, spatial memory, and seeking behavior, which lead to time consuming efforts to accomplish relatively straightforward tasks. Although this approach allows the participant to complete the task, it is effectively a workaround to an environment not explicitly designed for blind people [12]. Furthermore, not all blind people conceptualize their computing environments in the same way [38]. In general, most people, whether visually impaired or not, neither want nor find it easy to memorize commands, hence there is a downfall of DOS, UNIX, and other text and command-based systems in the mass market [27]. In our fieldwork, even when users were able to memorize and use commands, they behaved inconsistently depending upon the actions they performed. An application and document approach is a system-oriented rather than an activity-oriented perspective, and it continues to be reproduced by accessible systems that simply mirror existing structures. Treating commands and the objects they act upon as a single activity with consistent behavior across them would likely greatly improve experience for nonvisual computing users. In this case, the notion of the activity as an organizing structure becomes even more important relative to this need for consistent behavior. Additionally, when the system acts inconsistently, the challenges to receiving and understanding system feedback for nonvisual users exacerbates these issues, as we discuss in more detail in Section 4.3 below.

Issues of consistency within an activity are not limited to the actions taken upon the various services and data within the activity. The fundamental metaphors themselves can, at times, break down within a single activity when the activities require use of multiple pieces of software or other computational infrastructure. For example, during the same assignment as the vignette above, one student struggled to understand the difference between the folders created in the music application iTunes and the folders he had just created on the desktop. This kind of confusion is of course not limited to blind and low-vision computer users. However, without the visual feedback that drives this metaphor, the organizational underpinnings that differentiate a virtual folder (iTunes) and a physical memory-based folder (File System) make even less sense.

Finally, computer users often switch among documents, services, and applications within a single activity. A simple example might involve reading an email that asks for a meeting, switching to the calendar to check availability, and then switching back to the email client to respond. For sighted users, switching between windows is an expected and natural part of a multitasking environment that can quickly grow to an unmanageable state in complex multi-window environments. To support this growing complexity, systems have introduced new ways of visually arranging information such as tabs, split panes, and virtual desktops. For the nonvisual user, however, these visual conveniences are lost, instead introducing additional complexity. For example, the difference between a browser window and browser tab is conceptually insignificant for a nonvisual user, yet the behavior and interaction between the two require a different set of commands. Again from our field notes:

> The instructor often emphasizes the importance of learning how to use multiple browsers. As students were practicing browser commands later that day it became apparent how unnecessarily complicated this made switching from one website to another. Some students would open new browser windows to visit a different website, while others would open new tabs. One student asked me for help finding

> a webpage he had opened. The page he was looking for was open in a different browser application, yet he was using the key commands to cycle through open tabs. After a brief discussion, I realized that he had manually opened the Firefox browser to work on his assignment, then proceeded to select a link from his assignment, which opened the default browser (Internet Explorer) associated with hyperlinks in his system. – Field notes

Issues such as the one described in this vignette can be complicated further by skill and visual acuity. Although all of our participants were legally blind, several of them relied on their residual visual abilities to support their computer use. When we asked participants to open three websites using their personal computer during our experimental study, we observed nearly all of them completing the task in a different way. The behavior of one novice participant was noted in our study observations:

> When the task required her to switch from one web page to the next, she was unsure which keys to press to make the switch. The PI instructed her to press the alt+tab key combination. When it came time to move to the third web page, she used the same key command only to be surprised when she was returned to the first web page again. The PI explained that to move to the third web page she would need to press the tab key twice. – Observation from experimental study

This instance is likely a familiar experience to sighted-users of windowing interfaces, but without the visual cues displayed on screen to indicate which window is activated, nonvisual users are left to investigate further to resolve the breakdown between expectation and outcome. Here the participant expected a circular switching behavior where each key press moves to the next available window. If she had been switching between tabs, her expectation would have been met, but since she was switching between windows the behavior did not match her mental model. A different participant, an experienced low-vision computer user, explicitly stated his preference for browser tabs. Observing him carry out the web page setup task, he split duty between his screen magnifier, which kept the tabs enlarged in a narrow window at the top of his screen, and a handheld magnifying glass used to scan the content of the web page. This configuration simplified his interaction with the screen magnifier, constraining its use to horizontal movement, while he scanned the page using a physical magnifier. For this person, rather than contend with the complexities of maneuvering a magnifier around the screen, he opted to incorporate a physical tool to mediate task completion.

The challenges to organization of data, services, and applications for nonvisual users within a single activity indicate that existing computational abstractions are not sufficient for blind and low vision users. Systems in the existing research literature that have attempted to take an activity centered approach [4, 5, 18, 55, 64–66], rely on deep integration with the operating system or customized software to adapt the application-document model to activity. For nonvisual users, the adaptation layer already exists in the form of tools such as a screen reader or screen magnification software, which restructure computational information into a more suitable format (*e.g.,* speech, sonification, and magnified graphics). Therefore, introducing activity to a nonvisual system is a matter of rethinking how existing tools present computational information, rather than introduce additional complexity through new layers of software.

4.2 Activity Tracking

Activities tend to evolve over time as people change their goals and the objects they act upon (*e.g.,* a line of text in a document, image on a web page, or higher order element like an application)

and adapt to user actions. In computing environments, this behavior can result in the calling up and subsequent abandonment of a variety of documents, services, and applications. An object that played an important role at the start of an activity may never be used again. Similarly, some objects may not find utility until the final steps leading to activity completion. Managing these variations throughout the life of an activity is supported in the traditional desktop through visually oriented design cues such as recognition and spatial arrangement, and virtual desktops. The absence of these visual cues creates a variety of challenges to the orientation of nonvisual users within their applications and data [1, 11, 41]. Despite progress in the quality of screen readers and other accessibility tools, in our own fieldwork, we continued to see orientation within, tracking, and pausing of activities as significant challenges. In particular, students in the program we observed regularly struggled to restart after a pause, engaging in seeking behavior or simply restarting the entire activity from the beginning, as described in the following vignette from our field notes:

> The instructor had finished lecturing for the day and freed the students to start working on their assignments. I was observing one student who was working on a web page navigation task. After a brief hesitation she removed her headphones and asked the instructor for the command to open a new web page. Upon receiving a response, she entered the command "control-w," even though the instructor had told her "control-t," before putting her headphones back on. The action had mistakenly closed the browser, so when she proceeded with her task, the system did not respond as expected. When I asked her what happened, she told me that her computer was acting up again and probably needed to be restarted. – Field notes

As demonstrated in this example, the role that each artifact plays, and its use (or disuse) at various times are critical to contextualizing the activity, but without the system's awareness that an unattended action was executed, the user can easily get lost. These challenges are not unique to nonvisual users. In more complex activities, in particular, sighted users also struggle with task reconstruction. For example, while writing a paper, the author might simultaneously reference a spreadsheet in a separate window on the desktop. In the traditional application-document metaphor, from a system perspective, these two documents have no knowledge of each other; yet from the user's perspective they are core parts of the same activity, which she may express by laying them side by side in the visual computing environment. Once these applications are closed, the meaningful connection between the documents is lost, requiring the author to reconstitute the activity when edits are required. Thus, task reconstruction through activity tracking is a central element of various activity-based systems for sighted users. The Kimura system, for example, tracks the unique memory handles that the operating system assigns to application windows [65]. Whereas in Bardram's system, applications are built on top of a framework that manages system state throughout an activity [5]. In both cases, the changes that occur to applications and documents (i.e., opening, closing, focus, and location) are captured and logged into a data store, effectively providing users with the ability to navigate backward and forward throughout the activity lifecycle. The consequence of designing these systems around sighted activity, is that they must rely on visualizations to allow the user to interact with the history. Visualizations are often difficult to translate into audio, requiring descriptive text as well as tabular representations of data that can be traversed by a keyboard. The additional complexity that these additional steps introduce makes the activity tracking format used in these systems difficult to use in a nonvisual environment.

In the vignette above, the student missed a notification that she had inadvertently closed a window because she did not hear the auditory cue—a subtle, but critical failure resulting from the ephemerality of the audio interface that does not match to the temporal patterns of the user's

activities. Alternatively, an activity-centered system should support these activities—and the underlying cognitive behaviors required to maintain them—by tracking key contextual indicators and alerting the user at a time appropriate to the context of use and larger goals of the activity. Similarly, in an activity-centered system, the action of opening and closing web pages connects them through their use during the activity. By tracking and recording use, these changes can be reconstructed at any given moment throughout the life of the activity. Tracking can alleviate the challenges that surround interruption and error recovery for nonvisual users.

4.3 Operationalizing Actions

Leontiev organized human consciousness into a three level hierarchy of activity, action, and operation [42]. Activities are carried out through actions, determined by the individual goals of the human subject. Actions are fulfilled through unconscious operations, which reflect the human subject's natural attributes. The relationship between actions and operations is described as fluid, where actions can become unconscious operations through the natural internalization that occurs through practice, and operations become conscious actions through externalizing processes such as breakdowns [10]. In computational terms, we can conceptualize the relationship between action and operation by observing how one might learn to use a computer mouse. At first, mouse use might be action oriented, where a novice user *consciously* interacts with its various buttons and controls. Eventually, through practice, mouse use becomes operationalized, moving from a conscious to *unconscious* operational state. As an operation, focus is shifted from use of the mouse itself to performing actions that the mouse supports, only returning to an action when an attribute of the mouse changes (e.g., a broken button or dead battery). Proponents of activity-centered computing [4, 36, 65] have demonstrated that structuring computation around activity can lead to improved support for operationalizing actions. In practice, however, configuration and management is a common point of difficulty for users. Tasks integral to supporting activity, such as "tagging" were commonly avoided [64]. Similarly, parts of the systems that required users to change their pre-existing practices were met with resistance [4, 64]. By restructuring the application-document metaphor, activity-based systems are effectively adding an additional layer of complexity that users must navigate.

From a nonvisual interaction perspective, the negative effects of restructuring application to activity is not surprising. The process of translating graphical information to auditory information for blind and low-vision users is a significant factor in keeping nonvisual interaction at the level of conscious action [3, 41, 63]. Poor translation is both burdensome and inefficient, making it noticeably intrusive to participants in our field work. Students as well as their blind instructor continuously faced challenges with hardware and software. As we noted in our field notes:

> One low-vision student was using screen magnification software to perform the tasks being taught by the instructor. Over the course of the lecture, her computer became increasingly unresponsive, freezing for 15-30 seconds between actions. Unable to keep up with the rest of the class, she asked the instructor for assistance. Together they spent the rest of the class time attempting to solve the problem. – Field notes

Technical issues like the one captured here occur frequently. In our classroom fieldwork, these issues resulted in either time lost for the student, who was dealing with the problem, or for the entire class who had to wait for the instructor to resolve it. In a workplace, these issues can interfere with accomplishment of mission critical tasks in the worst case scenario or just an employee being less efficient or perceived to be more troublesome in the best case scenario [13]. In some cases, such issues cannot be easily resolved by the user or people nearby, leading to the entire system needing

to be set to the side until a specialist can engage. In our classroom-based fieldwork, the varying inconsistencies of the tools students were attempting to learn were a constant source of distraction.

Today I was asked to help one of the more advanced students in the class get her new Windows 8 computer setup with the Firefox and Chrome web browsers. I provided some verbal directions on where to go to download each browser, but otherwise left navigation and interaction to her. The Firefox browser downloaded and installed without issue, however, the Chrome installer interface was not detected by her screen reader, leaving her to conclude that the installation did not work. Although I could see that the installation was functioning properly by observing the installer progress bar, since her screen reader did not register the progress control, she was left without any feedback. –Field notes

The wildly different experiences while performing the same action can leave nonvisual users attempting to solve problems that do not actually exist. A common tactic we observed was to start over. In the example provided here, the first author was able to intervene by manually providing the required auditory feedback, preventing the student from following her instinct to start the download over again. In many other instances, however, students opted to reboot the computer. An extreme, but effective, resolution to the problem which was often viewed to be successful, not because it fixed anything, rather that it allowed students to reset their frame of reference within the system.

The difficulties we observed with screen reading and screen magnification tools stand in stark contrast to the students' use of a handheld audio recording and playback device called the Victor Reader.[6] Victor Readers were assigned to all the students in the class, allowing them to record lectures, take notes, and play back text files. Roughly the size of a small smartphone, the device relies on a small array of tactilely differentiated buttons instead of a screen. While the use of screen translation tools were continuously conscious interactions, students operated the Victor Reader without issue. They spoke positively about their interactions with the Victor Reader and made regular use of it. One student in particular made a habit of transferring all of her documents to the device to read, rather than relying on her desktop screen reader.

Use of the Victor Reader yields valuable insights into the frictionless interaction that nonvisual technology can provide when translation of a graphical interface is not a dependency. Realistically, nonvisual interaction with graphical systems simply is not possible without some amount of translation to alternative modalities. However, the model that is used can be shaped to fit the more natural practices of human activity, making the process less complex. For example, at the level of activity, there is no need to consider where a document is stored or how it is saved; those details are managed by the system. The benefit of this reduction in cognitive tasks grows with the complexity of activity, essentially unifying the actions of saving and storing across many documents into one single action. To make systems truly lead to unconscious operations for nonvisual users, however, they must be reconsidered in light of the activities they are meant to support. Shifting the translation of information between modalities from a model of individual elements within documents within applications and folder hierarchies to one that considers information flows across activities could enable this kind of invisibility in use for blind and low-vision users.

4.4 Summary

In this section, we have provided an activity theoretic analysis of computer use data collected from field work and an experimental study with blind and low vision participants.

[6] http://www.humanware.com/microsite/stream/index.html.

5 DISCUSSION

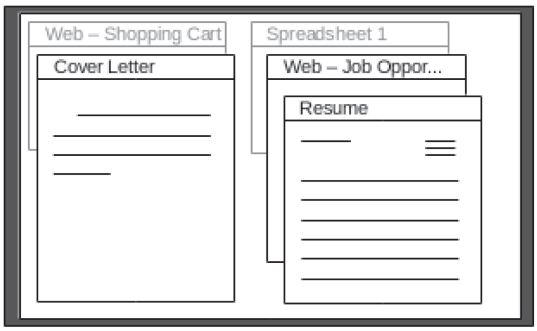
In this section, we reflect on our analysis with a discussion about how core concepts of activity theory can inform how we think about designing computational systems for nonvisual use. The operationalizing loop described in Section 4.3, the idea that we cease to "see" the tools we are using as they become a natural part of the interaction, frames much of what we view as the end goal for accessible tools. Current technologies, such as screen readers, assert themselves aggressively into the workflow of a nonvisual computer user. The white cane, on the other hand, is a dramatic alternative that is quite literally a physical extension of the blind user's arm. Framing nonvisual interaction by activity provides a mechanism by which we can raise the quality of the interaction of nonvisual digital objects to that of the cane, and operationalize interactions for blind users.

Prior research [34, 50] notes that true unconscious operation arises through the unique properties of both user and interaction. Thus, attempting to design for unconscious operation directly may be somewhat impractical. What we offer here, then, is not a prescribed set of requirements or implications for design. Rather, we propose multiple pathways to operational actions such that designers who restructure their interactions around the notion of activity can improve the potential for unconscious operations to emerge with time, particularly for nonvisual computer users for whom traditional approaches are noisy and interfering. Specifically, nonvisual systems should be redesigned from the ground up with a consideration for the fundamental goals of users rather than act as translators for the tasks sighted users might complete to accomplish those same goals. Second, a consideration for activity must not end with the initial goal but must be allowed to adapt and change over time. Finally, in addition to considering that blind users might engage different tasks than sighted users, an activity centered approach to nonvisual computing explicitly considers the way these tasks are carried out across applications. In this section, we describe the considerations designers must engage to merge nonvisual computing with the more natural orientation of human activity.

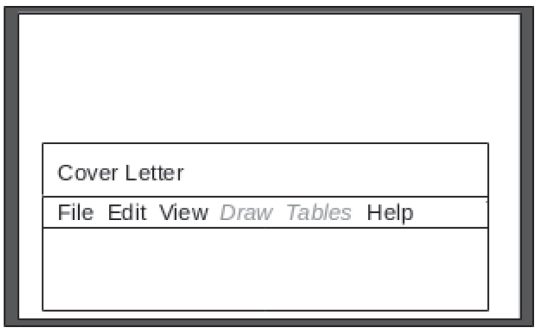
5.1 Tasks Vary by Activity, Interaction, and Ability

In an activity-centered model, attention flows as needed to the tools and tasks that make up the overall activity. Like the carpenter's work bench, when hammering a nail, the unneeded saw rests out of sight and mind. In a desktop computing environment, sighted users can ignore icons, windows, and content that do not relate to the task at hand. For the nonvisual user, however, screen translation tools do not filter for contextually relevant information, forcing unnecessary content into the processing stream. Filtering out unwanted information, to the extent it is possible, requires workarounds such as increased rate of speech and keyboard shortcuts. The problems that arise from these workarounds are well established [7, 24, 40, 41] as well as attempts at resolving them [15, 52, 62]. The issue an orientation towards activity highlights, however, goes beyond these workarounds and their challenges. Rather, the problem rests in the inability of the system to truly understand the activity, and therefore its sub-tasks. The system adheres to an interaction model that does not match user needs, the driving force behind human activity [34].

Existing computer hardware already has the power to track, model, and act on far more about the people using it than typically gets employed in practice. Even in much simpler systems, such as the Victor Reader described earlier, a more appropriate interaction model can present a rapid path towards unconscious operations. The Victor Reader works, in large part, because its task is not something that sighted computer users ever do. It was built from the beginning for a different interaction model, rather than retrofitted on top of an interaction model made for people with sight. Input techniques and their associated hardware for more fully featured systems should similarly be redesigned with a consideration for the specific needs, then tasks, then interactions of non-sighted users.



(a) Activity Level



(b) Application Level

Fig. 4. Use of activity hierarchy for nonvisual interfaces. (a) A typical desktop environment in which multiple applications are open. Since the user is currently focused on writing a cover letter, unrelated applications (*i.e.,* Shopping and Spreadsheet) are not included in the audio space. (b) The system is aware of the requirements to create a resume cover letter and automatically removes unnecessary application features (*i.e.,* Draw and Tables) from the audio space.

When interaction is modeled on need, comprehension is transferred from the user to the system. Intuitively, a user expects only certain features to accomplish a particular task. Rather then present all features equally, a system or tool should present only the features required to complete the task. Using the hierarchical structure of activity, we illustrate the concept of need through the following scenario:

5.1.1 At the Level of Activity. A user needs to write a cover letter for a resume. The nonvisual activity centered system recognizes this need as one that only requires a subset of the applications available on the system (Figure 4 (a)). For example, while working on a resume a user might require a word processor to write the letter, a copy of the resume, and a web browser to reference a job application. While the activity is active, from the perspective of the user, these three tools are the only computational entities available in the system.

5.1.2 At the Level of Application. The word processor that is being used to draft the cover letter, recognizes the need and limits its feature set to only interactions suitable for letter writing (Figure 4(b)). From the perspective of the user, features such as tables, drawings, images, and macros do not exist.

At both levels, the audio stream is isolated from unnecessary and unsolicited entities running on the system, thus narrowing the presentation of information to match user expectation. As the

scenario depicted in Figure 4 highlights, isolation does not remove entities from the environment, rather, they are simply not communicated according to the context of the current activity. Just as the sighted user chooses to ignore unrelated visual information, the isolated audio stream enables nonvisual users to ignore unrelated audio information. Likewise, as the sighted user can quickly shift attention to previously ignored entities, a nonvisual user can expand outside of the isolated context to support shifting needs. We see interaction models built upon user need as an opportunity for designers to remove the obstacles that delay unconscious operation. The natural binding between need and expectation enables greater flexibility in how computational entities are presented nonvisually.

5.2 Make Change a First Class Citizen

Activities are shaped by "virtue of their differing objects [51]." When applied to computational systems, the unique footprint that objects give to activity is commonly used as the basis for modeling activity centered interactions [5, 36, 64]. An activity-centered approach then emphasizes the ability to pause, port, and resume activities alongside the ability to understand what it is that the system is doing on behalf of the user. This emphasis requires that the user know what has changed in the interface while working on an activity, between moments of pause, and across multiple platforms. Unfortunately, as we saw in our fieldwork and as many researchers have noted (*e.g.*, [3, 14, 28, 49, 54]), audio is the primary mode of interaction for nonvisual computing users, and it is ephemeral and sequential and for the most part, individual and non-portable.

For most people, the visuospatial sketchpad is more capable of retaining extensive information than the phonological loop [2]. This increased retention enables users to quickly scan the computing context visually to gain a firm understanding of what has changed in the interface. For nonvisual users, if the interface is complex, comparing it to the prior version may require extensive listening, well beyond the capabilities of the phonological loop, and requiring a re-listening to the audio simply to orient oneself to the current system state.

Therefore, reconstructing the task to integrate change into the set of base interactions that a nonvisual user has available introduces a path for task reconstruction. Orienting a system towards activity reduces the complexity of managing change in an interaction model. The activity is, or at least could be, responsible for the state of all objects in use for task completion. Placing this responsibility on the activity removes it from the user, freeing them from having to manage system level tasks (*e.g.*, locate, open, and save). Furthermore, by assigning the role of object management to the system, a record of changes within the activity is captured, effectively generating a timeline of interaction events that support change. For the nonvisual user, change management represents a significant reduction in the amount of auditory information that must be processed during task completion.

5.3 Design for Goals Not Applications

As noted above, much of desktop computing for blind and low vision users is accomplished by translating the activities, tasks, and interactions of sighted users into a nonvisual paradigm. One of the primary limitations of this approach is the need to work across applications and the limited ability to translate visual information effectively in the multi-application computing environment. Because a nonvisual user might carry out a particular activity using a drastically different set of tasks than a sighted user might use, a nonvisual user is likely to make use of applications in different ways and possibly use different applications. Thus, any computing system built on the notion of activity would need to consider differences in task decomposition and its associated applications.

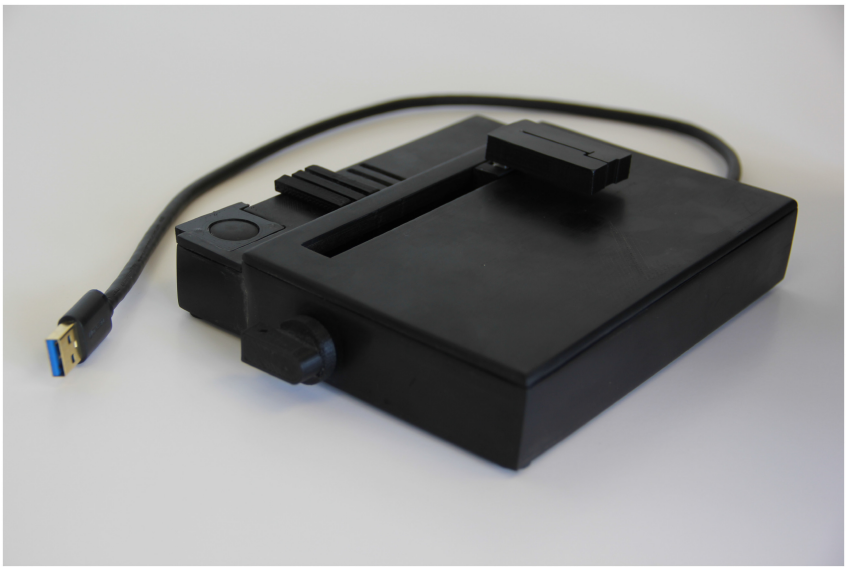


Fig. 5. The KInD. On the left side, the activity context dial is in the application content position. The thumb sits in the center of the device, ready to move to the left or right.

To speed up working with nonvisual computing systems, many advanced users memorize a variety of commands. Efficient computer use often only occurs upon mastering recall of dozens of commands. Even after reaching this level of mastery, however, users still have work to do any time a new application needs to be accessed. Commands can have similar actions across a variety of applications, different actions even for very similar applications, or simply be repurposed to meet the changing requirements of a single application. An activity centered approach would ensure that similar interactions map to similar tasks and have similar outcomes. A strict visual to auditory translation of existing interfaces cannot achieve this, but transformation of actions can, and designers of accessible systems should work to this ultimate end goal.

Activity serves as a natural extension for computer interaction by closely aligning with user expectation to support unconscious operations. An activity centered approach introduces ways for designers to address many of the known barriers in nonvisual computing, such as excessive audio, unexpected behaviors, and command memorization [3]. Solving these challenges alone does not completely fix the problems associated with nonvisual computing, such as accessibility compliance [45, 56]. Computing is ever more complex, requiring rethinking the entire interaction and architecture from the ground up for a truly accessible experience. Rather than adapt the interaction and tasks common to sighted users, activity theory suggests a different way of thinking about design for accessibility, one that provides this fundamental restructuring we suggest.

6 CASE STUDY

To explore how our activity theoretical design considerations can be applied to nonvisual computing, we developed an activity oriented application programming interface (API) on top of the Google Drive cloud based file management system [26]. Our API can be controlled through either a command line interface (CLI) or a custom device we created called the Kinesthetic Interaction Device (KInD) (see Figure 5). KInD is a tangible interaction device that shares many of the characteristics we introduced with the Tangible Desktop (see Section 3.2), but goes further by providing

richer haptics and increased portability. KInD can be programmed to support many different types of interactions that benefit from proprioceptive and tactile input and output. The activity API provides text-to-speech output to communicate information auditorily. In this section, we first describe the activity API, including an overview of its CLI and KInD interfaces. We then conclude with a discussion of our findings from two design sessions conducted to elicit feedback from blind and low-vision computers users.

6.1 Activity Interaction

The activity API overlays a set of activity-oriented interactions on top of Google Drive [26] to support a platform independent interface for nonvisual activity. To demonstrate one way that activity theoretical concepts can benefit nonvisual computing, our activity API implements a small subset of the types of interactions we imagine a complete nonvisual activity-centric platform might require. As we described earlier in this section, the API is accessed and manipulated through two complimentary interfaces, a CLI and a prototype computer peripheral, which we describe in detail in the next two sections.

6.1.1 Activity CLI. The Activity CLI provides a natural language point of interaction for the activity API. The CLI was designed to be used as a standalone interface or in conjunction with either the Google Drive web interface, or KInD (described in the next section). There are the following five primary commands interpreted by the CLI to control the API: create, move, list, tell, and help. The create and list commands are combined with additional parameters recognized by the API to perform operations. For example, the create command is combined with an application type to create a new file in Google Drive (*e.g.*, "create spreadsheet" or "create document"). The move command enables file movement from activity to another. The remaining commands are used to provide context about system state such as which activity is active, the applications within the current activity, and activities in the system.

6.1.2 KInD: Kinesthetic Interaction Device. Similar to our work on the Tangible Desktop [3], KInD relies on kinesthetic resistance, vibrotactile touch, and proprioception to translate input and output without audio descriptions. However, unlike the Tangible Desktop, which was distinctly designed to support an application-centric interaction model, the design and interaction model supported by KInD have been structured to support activity-centeric interaction. KInD makes use of three physical interactions that enable users to accomplish their goals nonvisually. First, KInD represents activities through tactilely differentiated tangible tokens to support computational organization in the physical world. Second, KInD tracks the systematic changes that occur during an activity, enabling a user to tangibly move between tasks temporally rather than through a method of auditory repetition used by traditional screen readers. Finally, KInD implements tangible spatial structure to encourage operationalization through proprioceptive memory rather than auditory seeking [63]. We describe each physical interaction in greater detail below.

Activity Management. Activity *tokens* (see Figure 5) are tangible icons that represent an activity. KInD only requires that a token be placed on it for that token's activity to become active. At that point, the associated documents and applications become available for use. An advantage of this approach is that non-active tokens can be physically arranged in whatever manner the user desires—allowing one's proprioceptive abilities to be leveraged for activity organization (*e.g.*, work on the left and entertainment on the right). Transferring the representation of activity to the physical world has the advantage of introducing the effects of unconscious operation that we previously described through classroom use of the Victor Reader (see Section 4.3). Like the uniquely shaped buttons on the Victor Reader, activity tokens can convey meaning through shape, size, and text. For example, the sample tokens created for our platform could be 3D printed with

Table 2. Table of Activity Contexts

Activity Contexts	
Context Level	*Description*
1	Activities
2	Applications and Documents
3	Application Menus
4	Application Content
5	Activity History

braille text or different shapes, allowing tokens to be molded to the mnemonic strengths of the individual, reducing the burden of command recall.

Interaction History. KInD is situated between the intentions of the user and the execution of those intents in the computer system, allowing our infrastructure to capture input and output events as they occur. Events are logged in the host computer's file system to provide a history of interaction, and associations between activity token and entities. The primary responsibility of the interaction history is to support the dynamic nature of an activity. As system entities are opened they become an active part of the activity. When they are closed, they move out of the activity space, but remain in its history. This allows the platform to restore the last known state of an activity upon activation. Furthermore, it introduces a historical archive of the activity that can be used for retracing steps and error recovery (*e.g.,* accidentally closing a document).

Contextual Change. To accommodate the movement across activities as well as their associated entities, KInD functions within different contexts (see Table 2). The top context sits at the top level of an activity, supporting the movement of entities in and out of an active activity. The next context functions within an activity, providing support for movement across entities associated with the activity. The third context supports control at the application level, allowing KInD to be used to traverse and select features of the active application. Finally, the fourth context supports movement across the content of the application (*e.g.,* traversing a web page or document). Switching between contexts is managed by a physical rotary dial on the left side of KInD. The dial uses detents and audible confirmation to indicate when it has been rotated into a new context. As with the activity tokens, the physical manipulation enables proprioceptive abilities to quickly identify the desired context.

Virtual Docks. Organization within an activity, potentially involving tens of individual entities (*e.g.,* application file, email, or web page), is too complex to be handled through physical manipulation of an activity token. Instead KInD implements virtual docks to sort and store the entities assigned to an activity. Docks are virtual, in that they are managed through a software subsystem, but are accessed through physical interaction using the KInD slide bar. As new entities are added to an activity a new dock is created. A haptic detent is generated by the slide potentiometer to indicate traversal across an entity. So if an activity contains two entities, a detent is felt at the midpoint of the slide traversal, with three entities, two detents are felt equally divide along the slide, and so on. Individual entities can be arranged within these slots in a manner most suitable for the user. For example, a user might prefer to keep web pages in the right most slots, a presentation document in the far left, and everything else in between, retaining the advantages of proprioceptive recall found in activity tokens. Just as the flow of work can be dynamic, entities can flow in and out of activities depending on user need. For example, a user might want to take a break from a work activity and browse the news, a goal preferred not be associated with the work activity. KInD supports this type of activity flow by enabling docked entities to be pinned outside

the activity specific scope. Pinning enables individual entities to be moved between activities or simply detached from an activity entirely.

6.2 Mapping KInD Features to Activity-Centered Design Considerations

In this section, we map the functions of the activity API and KInD to the design considerations discussed in Section 5. We start with a brief summary of each design consideration followed by a description of how it is supported by the activity API, KInD, or both.

6.2.1 Modeling Interactions on Need. In Section 5.1, we describe how an activity-centered nonvisual system should consider user needs, tasks, and interactions to reduce complexity in a nonvisual computing environment. The *Activity Tokens* and underlying API in KInD support user need by organizing interrelated computational entities into high-level activities. Activity Tokens can be interchanged as user needs shift, keeping the working environment isolated from unrelated entities.

6.2.2 Supporting System Change. In Section 5.2, we demonstrate the importance of tracking changes within a computing environment and how an activity-centered interaction model introduces additional facilities for supporting change. The KInD system supports change in two ways. First, by providing tactilely unique representations of activities through *Activity Tokens* and activity context through *Virtual Docks* and the *Context Dial*, users can directly recall system state. Second, *Interaction History* enables users to reconstruct the changes that occurred to an activity over time. Collectively, these features of KInD serve to reduce the number of interactions a nonvisual user must memorize and execute to manage system change. For example, removing an activity token implicitly signals to the system that user goals have changed, enabling state management tasks (*e.g.,* pause, save, or close) to be initiated.

6.2.3 Supporting User Goals. In Section 5.3, we discuss how the hierarchical nature of activity supports consistency of commands across computer applications. In the same way the KInD features collectively support change, they also work together to prioritize goals over applications. *Activity Tokens* and *Virtual Docks* remove much of the burden that accompanies file management tasks for nonvisual users by replacing multiple keyboard commands (many of which vary across applications [8]) with tangible interactions.

6.3 Exploring the Activity API through Design Sessions

To understand how our activity theoretical insights for accessible computing might benefit nonvisual users, we conducted design sessions lasting 2 hours per participant at a blind children's school (n = 4) and on our campus (n = 2) (see Table 4). We organized our design sessions into four phases to probe participant reaction to our activity-centered interaction model. The phases had short (less than 5 minutes) breaks between them, and all participants stayed for the entire session, including all four phases. In phase one (typically 30 minutes), participants were asked to rate their level of experience with computers (*e.g.,* novice, proficient, or expert) and describe the types of tasks they use their computer to complete (*e.g.,* taking minutes during a meeting). During the break between phase one and two, the research team oriented the remaining phases to the specific tasks described by participants during the first segment. In phases two–four (typically 15–20 minutes each), participants were asked to carry out a series of file management tasks (see Table 3) using the Google Docs web interface, our CLI, and KInD, respectively. Phase four began with a brief training period (typically 5 minutes), where the research team walked through KInD's features (see Figure 6) with the participant. Participants were encouraged to interact with KInD, to familiarize themselves with the tactile, haptic, and proprioceptive features utilized during task

Table 3. Design Session Tasks

	Task	*Category*
1.	Place a token on KInD.	Task Completion
2.	Using the context dial, select application context.	Change
3.	Open a document. Then open spreadsheet.	Task Completion
4.	Move context dial to activity.	Change
5.	Move document to the right side.	Change
6.	Remove the current token, place a new token.	Task Completion
7.	Move context dial to application.	Change
8.	Open email. Then open spreadsheet.	Task Completion
9.	Move context dial to activity, explore.	
10.	Move context dial to application, explore. (switching mode)	
11.	Pin email.	Change
12.	Switch token	
13.	Unpin document.	Change
14.	Move slider to find email.	Change
15.	Return to original document.	Change

Table 4. Participant Detail for the Case Study

		Participant Data			
ID	*Session*	*Experience*	*Gender*	*Visual Impairment*	*Age*
P1	1	Proficient	Male	Low Vision	35
P2	1	Expert	Male	Low Vision	43
P3	2	Proficient	Female	Blind	28
P4	2	Expert	Female	Blind	30
P5	2	Expert	Female	Blind	29
P6	2	Proficient	Female	Blind	30

The session column corresponds to the design session the participant attended. Session 1 was conducted at our university lab. Session 2 was conducted at a blind children's school. Participant experience level was self-rated (*novice, proficient, and expert*) during the first phase of the session.

completion. Once each participant acknowledged they were comfortable with KInD's functions, each file management task was read aloud by the research team as the participant performed the task. After the fourth phase was complete, a semi-structured discussion with participants was conducted (typically 20 minutes). The rest of our discussion will focus on the themes that emerged from the design sessions.

The design sessions provided useful insights into the structure and design of our activity platform. Overall, participants responded positively to the CLI and KInD interfaces to our API. While all participants moved fluidly through task completion, each expressed a unique perspective on their experience, ranging from indifference (P6) to enthusiasm (P2 and P3). P6, who primarily uses her computer for audio editing, admitted that her specific use case made it difficult to imagine how orienting towards activity would change her workflow. Whereas P3, whose job required her to record and manage a variety of different meeting notes, found the activity structure to be a positive change from her existing practices, comparing and contrasting the activity platform with her typical workflow. Other participants adopted similar patterns to explain how they thought the CLI

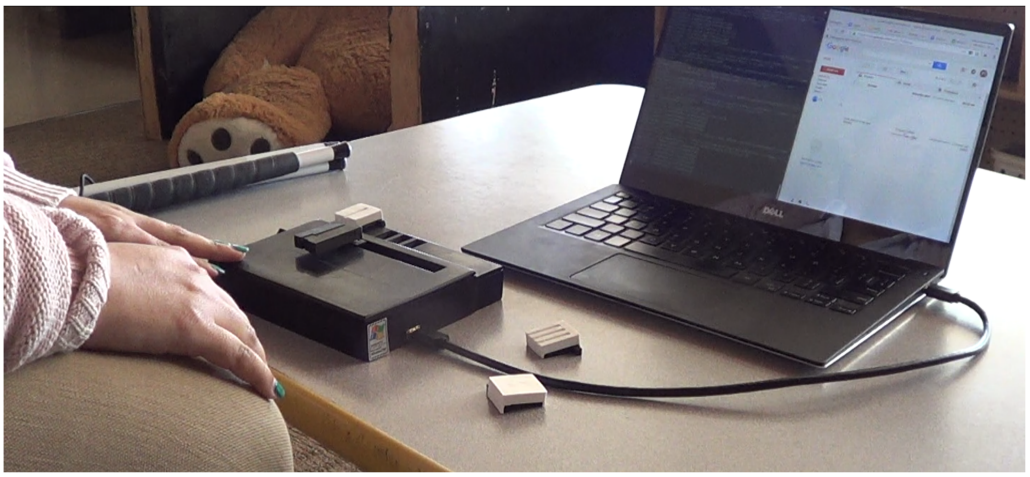


Fig. 6. An example of the setup used during design sessions with KInD. Activity Tokens have been positioned near KInD by the participant for quick retrieval.

or KInD might improve their own task completion. For example, P1 pointed out that he prefers to use his smartphone over desktop for most computing tasks due to its simplified interaction model, yet explained that KInD would make it easier for him to engage with his daughter on their desktop computer. P4 and P5 expanded on P1's experiences by explaining that the spatial movements used by KInD to manipulate activities aligned more closely with their smartphone interaction patterns. Similarly, P3, P4, and P5 described their challenges with command memorization and content traversal, noting that the natural language of the CLI interface (P3), and the tactile cues of KInD (P4, P5) could serve to lower dependency on command memorization.

Interestingly, P2, P4, and P5, who self-rated as expert computer users, also expressed benefits of the activity platform in terms of improvement for *other* users in addition to themselves. P2 articulated similar thoughts, envisioning the activity platform as an onboarding tool. He stated that although the feature set was small, its constrained options could help prevent users from getting lost. P1 agreed stating that he found KInD less intimidating than his desktop computer. These types of responses evoke sentiments of sociality similar to those observed by Morrison et al. They view their own experiences with desktop computing to be sufficient, but recall their own difficulties in learning, as well as the challenges faced by others, as problematic enough to desire an alternative.

As a preliminary investigation into the application of the activity theoretical insights we have described in this work, our case study provides one potential path forward. Although the informal structure of our design session prevents us from drawing concrete conclusions about the effectiveness of an activity-centered interaction model, we found the varied responses to the perceived simplicity of the system compelling. Participants viewed the activity system as both useful and simplistic enough to support their own work as well as the work of novice users, aligning with the action–operation process we discuss in Section 4.3. Therefore, we see a nonvisual model oriented towards activity as one that supports a constrained interaction space, but does not prevent more experienced users from accessing the lower level system. Just as the CLI and KInD provide a constrained interaction space on top of Google Drive, the full featured web based interface remains available for users who need or prefer it.

7 LIMITATIONS AND FUTURE WORK

We view this work as a first step towards adopting new ways of thinking about how nonvisual computing systems might be designed to lower the barriers to entry for blind and low vision computer users. However, we also acknowledge the limitations of the data used in our analysis. The field work and experimental data that informed our activity-centered design considerations are largely derived from novice computer users. While our findings suggest that activity-centered interaction can remove some of the obstacles that users experience in nonvisual computing environments, an activity analysis of the practices of more advanced users could reveal additional insights. Despite the emphasis on novice computer use in our data, we do see similar experiences with intermediate and advanced users elsewhere in the literature. For example, Billah et al. identified technical challenges similar to those we observed including inconsistencies across software tools and operating systems that forced users to learn multiple paths towards completing a task [8]. Additionally, Potluri et al. identify discoverability and navigation as two high-level challenges blind and low-vision programmers encounter with their tools [53]. Both Billah et al. and Potluri et al.'s work included participants from a range of professions and skill levels, indicating that the challenges we found manifest in similar ways for visually impaired users, regardless of skill level. Furthermore, the diversity of efforts to uncover workarounds to desktop computing and ICT more broadly, including those we review in Section 2.1, highlight the value of rethinking the nonvisual computing experience.

In future work, we aim to test the efficacy of the approach presented in our case study through continued design and evaluation of activity-centered nonvisual interaction tools like KInD. We are particularly interested in learning how assistive devices like KInD can support operationalizing routine computer interactions. Finally, as we discussed at the beginning of this section, further research is needed to understand how intermediate and expert behaviors might influence the design of activity-oriented interactions.

8 CONCLUSION

In this work, we used an activity theory lens to analyze nonvisual computing for blind and low-vision users. Our analysis indicated major challenges for users in translating the activities they were working towards into specific tasks to be completed in a system comprehensible manner. Specifically, blind and low-vision students learning to use accessible technologies struggled with organizing their activities, tracking the history and status of their operations, and understanding how the system was acting underneath these interactions.

We demonstrated how principles of activity theory can be used to rethink the design of nonvisual computational systems. Organizing the visual to nonvisual translation of information around activity presents myriad ways to support operationalizing user interaction. In particular, designing to support the individual needs and motives of the user introduces more natural ways to present computational information. Designing clear, systematic distinctions between levels of activity interjects points of change that can be tracked to support user awareness within a system. And finally, transferring the burden of file and application management from the user to the system can reduce the overhead of task completion.

Activity theory helps us to see the ways in which the human activity and the system objects are co-constructed, updating and evolving alongside one another, and the reliance that the system has on people being sighted to make these intertwined interactions work relatively seamlessly. Activity-centered computing has been previously proposed as a solution to numerous mismatches between user activities and system behavior and as a more general approach for social analysis and design in HCI [35]. However, what we see in this work is how much more important this approach

could be for nonvisual computer users. Blind and low-vision users must wrestle with challenging workarounds to interact with sighted computational systems. Although the tools exist to support nonvisual interaction, they do not go far enough to resolve the imbalance with sighted users.

The work we have presented is heavily focused upon the desktop computing paradigm, which in recent years has experienced increased competition from other forms of computation such as smartphones, tablets, and voice assistants. As we demonstrated in our case study, there are instances when the touch interfaces of a smartphone or tablet can meet the needs of nonvisual users more effectively than a desktop computer. Through an activity theoretical lens, we can look beyond the individual application oriented tasks of each computational system and think about how each system can serve activities more broadly. We see this paradigm shift as an opportunity for designers and researchers to re-imagine how computation can support the activities of blind and low vision users.

ACKNOWLEDGMENTS

This work would not have been possible without the generous collaboration with Empowertech. This research was conducted under ethics protocol HS2014-1598 at UC Irvine. Finally, the authors thank the STAR Group at UCI, and the anonymous reviewers for their insightful feedback on this article.

REFERENCES

[1] Ali Abdolrahmani and Ravi Kuber. 2016. Should I trust it when I cannot see it?: Credibility assessment for blind web users. In *Proceedings of the 18th International ACM SIGACCESS Conference on Computers and Accessibility*. ACM, New York, NY, 191–199. DOI: https://doi.org/10.1145/2982142.2982173

[2] Alan Baddeley. 1992. Working memory and conscious awareness. In *Theories of Memory*. Lawrence Erlbaum Associates, 11–20.

[3] Mark S. Baldwin, Gillian R. Hayes, Oliver L. Haimson, Jennifer Mankoff, and Scott E. Hudson. 2017. The tangible desktop: A multimodal approach to nonvisual computing. *ACM Transactions on Accessible Computing* 10, 3 (2017), 9.

[4] Jakob E. Bardram, Jonathan Bunde-Pedersen, and Mads Soegaard. 2006. Support for activity-based computing in a personal computing operating system. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. ACM, 211–220.

[5] Jakob E. Bardram and E.2005. Activity-based computing: Support for mobility and collaboration in ubiquitous computing. *Personal and Ubiquitous Computing* 9, 5 (Sep 2005), 312–322. DOI: https://doi.org/10.1007/s00779-004-0335-2

[6] O. Bau, Poupyrev, A. Israr, and Harrison. 2010. TeslaTouch: Electrovibration for touch surfaces. In *Proceedings of the 23rd annual ACM Symposium on User Interface Software and Technology*. 283–292. DOI: https://doi.org/10.1145/1866029.1866074

[7] Edward C. Bell and Natalia M. Mino. 2015. Employment outcomes for blind and visually impaired adults. *Journal of Blindness Innovation and Research* 5 (2015). https://doi.org/10.5241/5-85

[8] Syed Masum Billah, Vikas Ashok, Donald E. Porter, and I. V. Ramakrishnan. 2017. Ubiquitous accessibility for people with visual impairments: Are we there yet? In *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems*. ACM, New York, NY, 5862–5868. DOI: https://doi.org/10.1145/3025453.3025731

[9] Meera M. Blattner, Denise A. Sumikawa, and Robert M. Greenberg. 1989. Earcons and icons: Their structure and common design principles. *Human–Computer Interaction* 4, 1 (1989), 11–44.

[10] Susanne Bødker. 1990. *Through the Interface: A Human Activity Approach to User Interface Design*. L. Erlbaum Associates Inc., Hillsdale, NJ.

[11] Yevgen Borodin, Jeffrey P. Bigham, Glenn Dausch, and I. V. Ramakrishnan. 2010. More than meets the eye: A survey of screen-reader browsing strategies. ACM, 13.

[12] L. H. Boyd. 1990. The graphical user interface crisis: Danger and opportunity. Retrieved from http://eric.ed.gov/?id=ED333687.

[13] Stacy M. Branham and Shaun K. Kane. 2015. The invisible work of accessibility: How blind employees manage accessibility in mixed-ability workplaces. In *Proceedings of the 17th International ACM SIGACCESS Conference on Computers & Accessibility*. ACM, 163–171.

[14] Stephen A. Brewster, Peter C. Wright, and Alistair D. N. Edwards. 1993. An evaluation of earcons for use in auditory human-computer interfaces. In *Proceedings of the INTERACT'93 and CHI'93 Conference on Human Factors in Computing Systems*. 222–227. DOI: https://doi.org/10.1145/169059.169179

[15] Andy Brown and Simon Harper. 2013. Dynamic injection of WAI-ARIA into web content. In *Proceedings of the 10th International Cross-Disciplinary Conference on Web Accessibility*. DOI: https://doi.org/10.1145/2461121.2461141

[16] Verena Cimarolli and Shu-wen Wang. 2006. Differences in social support among employed and unemployed adults who are visually impaired. *Journal of Visual Impairment & Blindness* 100 (2006), 545–556. https://doi.org/10.1177/0145482X0610000906

[17] Torkil Clemmensen, Victor Kaptelinin, and Bonnie Nardi. 2016. Making HCI theory work: An analysis of the use of activity theory in HCI research. *Behaviour & Information Technology* 35, 8 (2016), 608–627.

[18] Victor Cornet, Stephen Voida, and Richard J. Holden. 2018. Activity theory analysis of heart failure self-care. *Mind, Culture, and Activity* 25, 1 (2018), 22–39.

[19] Kai Crispien, Klaus Fellbaum, Anthony Savidis, and Constantine Stephanidis. 1996. A 3D-auditory environment for hierarchical navigation in non-visual interaction. Georgia Institute of Technology.

[20] Hilko Donker, Palle Klante, and Peter Gorny. 2002. The design of auditory user interfaces for blind users. In *Proceedings of the 2nd Nordic Conference on Human-computer Interaction*. 149–155. DOI: https://doi.org/10.1145/572020.572038

[21] Alo Dutta, Robert Gervey, Fong Chan, Chih-Chin Chou, and Nicole Ditchman. 2008. Vocational rehabilitation services and employment outcomes for people with disabilities: A United States study. *Journal of Occupational Rehabilitation* 18, 4 (2008), 326–334. DOI: https://doi.org/10.1007/s10926-008-9154-z

[22] W. Keith Edwards, Elizabeth D. Mynatt, and Kathryn Stockton. 1994. Providing access to graphical user interfaces–not graphical screens. ACM, 47–54.

[23] Kenneth P. Fishkin. 2004. A taxonomy for and analysis of tangible interfaces. *Personal and Ubiquitous Computing* 8, 5 (2004), 347–358. DOI: https://doi.org/10.1007/s00779-004-0297-4

[24] Kevin D. Frick, Emily W. Gower, John H. Kempen, and Jennifer L. Wolff. 2007. Economic impact of visual impairment and blindness in the united states. *Archives of Ophthalmology* 125, 4 (2007), 544. DOI: https://doi.org/10.1001/archopht.125.4.544

[25] William W. Gaver. 1989. The SonicFinder: An interface that uses auditory icons. *Human-Computer Interaction* 4, 1 (March 1989), 67–94. DOI: https://doi.org/10.1207/s15327051hci0401_3

[26] Google. 2019. Google Drive: Free Cloud Storage for Personal Use. Retrieved from https://www.google.com/drive.

[27] Jonathan Grudin. 2008. A moving target: The evolution of HCI. In *The Human-computer Interaction Handbook: Fundamentals, Evolving Technologies, and Emerging Applications*. 1–24.

[28] J. Guerreiro and D. Gonçalves. 2014. Text-to-speeches: Evaluating the perception of concurrent speech by blind people. In *Proceedings of the 16th International ACM SIGACCESS Conference on Computers and Accessibility*. 169–176. DOI: https://doi.org/10.1145/2661334.2661367

[29] Sidhant Gupta, Dan Morris, Shwetak N. Patel, and Desney Tan. 2013. AirWave: Non-contact haptic feedback using air vortex rings. In *Proceedings of the 2013 ACM International Joint Conference on Pervasive and ubiquitous Computing*. 419–428. DOI: https://doi.org/10.1145/2493432.2493463

[30] William C. Hill, James D. Hollan, Dave Wroblewski, and Tim McCandless. 1992. Edit wear and read wear. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. ACM Press, New York, New York, 3–9. DOI: https://doi.org/10.1145/142750.142751

[31] Humanware. 2018. Humanware - Brailliant Braille Displays - Blindness - Low Vision Aids for Macular Degeneration. Retrieved from https://store.humanware.com/hus/blindness/brailliant-braille-displays.

[32] Hiroshi Ishii and Brygg Ullmer. 1997. Tangible bits: Towards seamless interfaces between people, bits and atoms. In *Proceedings of the ACM SIGCHI Conference on Human Factors in Computing Systems*. ACM, New York, NY, 234–241. DOI: https://doi.org/10.1145/258549.258715

[33] Yvonne Jansen, Thorsten Karrer, and Jan Borchers. 2010. MudPad : Tactile feedback and haptic texture overlay for touch surfaces. In *ACM International Conference on Interactive Tabletops and Surfaces*. 11–14. DOI: https://doi.org/10.1145/1936652.1936655

[34] Victor Kaptelinin and Bonnie A. Nardi. 2006. *Acting with Technology: Activity Theory and Interaction Design*. MIT Press.

[35] Victor Kaptelinin, Bonnie A. Nardi, and Catriona Macaulay. 1999. Methods & tools: The activity checklist: A tool for representing the "space" of context. *Interactions* 6, 4 (1999), 27–39.

[36] Victor Kaptelinin and Victor. 2003. UMEA. In *Proceedings of the Conference on Human Factors in Computing Systems*. ACM Press, New York, New York, 353. DOI: https://doi.org/10.1145/642611.642673

[37] Roberta L. Klatzky and Susan J. Lederman. 2003. Touch. In *Handbook of Psychology*. Retrieved from http://onlinelibrary.wiley.com/doi/10.1002/0471264385.wei0406/full.

[38] Sri Hastuti Kurniawan and Alistair Sutcliffe. 2002. *Mental Models of Blind Users in the Windows Environment*. Springer, Berlin, Heidelberg, 568–574. DOI: https://doi.org/10.1007/3-540-45491-8_109

[39] Stacey Kuznetsov, Anind K. Dey, and Scott E. Hudson. 2009. The effectiveness of haptic cues as an assistive technology for human memory. In *Lecture Notes in Computer Science*. 168–175. DOI: https://doi.org/10.1007/978-3-642-01516-8_12

[40] Jonathan Lazar, Aaron Allen, Jason Kleinman, and Chris Malarkey. 2007. What frustrates screen reader users on the web: A study of 100 blind users. *International Journal of Human-Computer Interaction* 22, 3 (May 2007), 247–269. DOI: https://doi.org/10.1080/10447310709336964

[41] Jonathan Lazar, Jinjuan Feng, and Aaron Allen. 2006. Determining the impact of computer frustration on the mood of blind users browsing the web. In *Proceedings of the 8th International ACM SIGACCESS Conference on Computers and Accessibility*. 149–156. DOI: https://doi.org/10.1145/1168987.1169013

[42] Aleksei Nikolaevich Leont'ev. 1978. *Activity, consciousness, and personality*. Prentice Hall

[43] Alexej N. Leontjev. 1981. Problems of the development of the mind. Progress Publishers.

[44] V. Lévesque. 2005. Blindness, technology and haptics. *Center for Intelligent Machines*. McGill University, 22. DOI: https://doi.org/10.1.1.73.3807

[45] Jennifer Mankoff, Holly Fait, and Tu Tran. 2005. Is your web page accessible?: A comparative study of methods for assessing web page accessibility for the blind. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. ACM, 41–50.

[46] Muhanad S. Manshad, Enrico Pontelli, and Shakir J. Manshad. 2012. Trackable interactive multimodal manipulatives: Towards a tangible user environment for the blind. In *Proceedings of the International Conference on Computers for Handicapped Persons*. Springer, 664–671.

[47] M. C. McDonnall and A. Crudden. 2009. Factors affecting the successful employment of transition-age youths with visual impairments. *Journal of Visual Impairment & Blindness* 103, 6 (2009), 329–341.

[48] Cecily Morrison, Edward Cutrell, Anupama Dhareshwar, Kevin Doherty, Anja Thieme, and Alex Taylor. 2017. Imagining artificial intelligence applications with people with visual disabilities using tactile ideation. In *Proceedings of the 19th International ACM SIGACCESS Conference on Computers and Accessibility*. ACM, New York, NY, 81–90. DOI: https://doi.org/10.1145/3132525.3132530

[49] Elizabeth Mynatt. 1997. Transforming graphical interfaces into auditory interfaces for blind users. *Human-Computer Interaction* 12, 1 (1997), 7–45. DOI: https://doi.org/10.1207/s15327051hci1201&2_2

[50] Bonnie A. Nardi. 1996. *Context and Consciousness: Activity Theory and Human-computer Interaction*. MIT Press.

[51] Bonnie A. Nardi. 1996. Studying context: A comparison of activity theory, situated action models, and distributed cognition. In *Context and Consciousness: Activity Theory and Human-computer Interaction*.

[52] Peter Plessers, Sven Casteleyn, Yeliz Yesilada, Olga De Troyer, Robert Stevens, Simon Harper, and Carole Goble. 2005. Accessibility: A web engineering approach. In *Proceedings of the 14th International Conference on World Wide Web (WWW'05)*. Association for Computing Machinery, New York, NY, USA, 353–362. https://doi.org/10.1145/1060745.1060799

[53] Venkatesh Potluri, Priyan Vaithilingam, Suresh Iyengar, Y. Vidya, Manohar Swaminathan, and Gopal Srinivasa. 2018. CodeTalk: Improving programming environment accessibility for visually impaired developers. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems*. ACM, New York, NY, Article 618, 11 pages. DOI: https://doi.org/10.1145/3173574.3174192

[54] Dan Ratanasit and Melody M. Moore. 2005. Representing graphical user interfaces with sound: A review of approaches. *Journal of Visual Impairment and Blindness* 99, 2 (2005), 69–84.

[55] Tye Rattenbury and John Canny. 2007. CAAD. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. ACM Press, New York, New York, 687. DOI: https://doi.org/10.1145/1240624.1240731

[56] John T. Richards, Kyle Montague, and Vicki L. Hanson. 2012. Web accessibility as a side effect. In *Proceedings of the 14th International ACM SIGACCESS Conference on Computers and Accessibility*. ACM, 79–86.

[57] Brigitte Röder, Frank Rösler, and Helen J. Neville. 2001. Auditory memory in congenitally blind adults: A behavioral-electrophysiological investigation. *Cognitive Brain Research* 11, 2 (2001), 289–303. DOI: https://doi.org/10.1016/S0926-6410(01)00002-7

[58] Valkyrie Savage, Xiaohan Zhang, and Björn Hartmann. 2012. Midas: Fabricating custom capacitive touch sensors to prototype interactive objects. In *Proceedings of the 25th annual ACM Symposium on User Interface Software and Technology*. ACM, 579–588. Retrieved from http://dl.acm.org/citation.cfm?id=2380189.

[59] Anthony Savidis and Constantine Stephanidis. 1995. Building non-visual interaction through the development of the rooms metaphor. In *Proceedings of the Conference Companion on Human Factors in Computing Systems*. ACM Press, New York, New York, 244–245. DOI: https://doi.org/10.1145/223355.223562

[60] Bertrand Schneider, James M. Wallace, Paulo Blikstein, and Roy Pea. 2013. Preparing for future learning with a tangible user interface: The case of neuroscience. *IEEE Transactions on Learning Technologies* 6, 2 (2013), 117–129.

[61] R. Sodhi, Poupyrev, M. Glisson, and A. Israr. 2013. AIREAL: Interactive tactile experiences in free air. *ACM Transactions on Graphics* 32, 4 (2013), 134. DOI: https://doi.org/10.1145/2461912.2462007

[62] Markel Vigo and Simon Harper. 2013. Challenging information foraging theory: Screen reader users are not always driven by information scent. In *Proceedings of the 24th ACM Conference on Hypertext and Social Media*. ACM, 60–68.
[63] M. Vigo and S. Harper. 2013. Coping tactics employed by visually disabled users on the web. *International Journal of Human-Computer Studies* 71, 11 (2013), 1013–1025.
[64] S. Voida and E. D. Mynatt. 2009. It feels better than filing: Everyday work experiences in an activity-based computing system. In *Proceedings of the SIGCHI Conference on HumanFactors in Computing Systems*. Retrieved from http://dl.acm.org/citation.cfm?id=1518744.
[65] S. Voida, E. D. Mynatt, and B. MacIntyre. 2007. Supporting activity in desktop and ubiquitous computing. In *Beyond the Desktop*.
[66] Stephen Voida, Elizabeth D. Mynatt, and W. Keith Edwards. 2008. Re-framing the desktop interface around the activities of knowledge work. In *Proceedings of the 21st Annual ACM Symposium on User Interface Software and Technology*. ACM Press, New York, New York, 211. DOI: https://doi.org/10.1145/1449715.1449751
[67] Lev Semenovich Vygotsky. 1980. *Mind in Society: The Development of Higher Psychological Processes*. Harvard University Press.
[68] Bruce N. Walker, A. Nance, and Jeffrey Lindsay. 2006. Spearcons: Speech-based earcons improve navigation performance in auditory menus. In *Proceedings of the International Conference on Auditory Display*. 95–98.
[69] Mark Weiser. 1991. The computer for the 21st century. *Scientific American* 265, 3 (1991), 94–104.
[70] Malte Weiss, Simon Voelker, Jan Borchers, and Chat Wacharamanotham. 2011. FingerFlux: Near-surface haptic feedback on tabletops. In *Proceedings of the 24th Symposium on User Interface Software and Technology*. 615–620. DOI: https://doi.org/10.1145/2047196.2047277
[71] Malte Weiss, Julie Wagner, Yvonne Jansen, Roger Jennings, Ramsin Khoshabeh, James D. Hollan, and Jan Borchers. 2009. SLAP widgets: Bridging the gap between virtual and physical controls on tabletops. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. ACM, New York, NY, 481–490. DOI: https://doi.org/10.1145/1518701.1518779
[72] Oren Zuckerman and Ayelet Gal-Oz. 2013. To TUI or not to TUI: Evaluating performance and preference in tangible vs. graphical user interfaces. *International Journal of Human Computer Studies* 71, 7–8 (2013), 803–820. DOI: https://doi.org/10.1016/j.ijhcs.2013.04.003

Received July 2018; revised October 2019; accepted November 2019