Modality-Independent Interaction Framework for Cross-Disability Accessibility

J. Bern Jordan and Gregg C. Vanderheiden

University of Wisconsin–Madison, Madison, Wisconsin, United States
{jordan,gv}@trace.wisc.edu

Abstract. People with disabilities often have difficulty using ICT and similar technologies because of a mismatch between their needs and the requirements of the user interface. The wide range of both user abilities and accessibility guidelines makes it difficult for interface designers who need a simpler accessibility framework that still works across disabilities. A modality-independent interaction framework is proposed to address this problem. We define modality-independent input as non-time-dependent encoded input (such as that from a keyboard) and modality-independent output as electronic text. These formats can be translated to provide a wide range input and output forms as well as support for assistive technologies. We identify three interfaces styles that support modality-independent input/output: command line, single-keystroke command, and linear navigation interfaces. Tasks that depend on time, complex-path, simultaneity, or experience are identified as providing barriers to full cross-disability accessibility. The framework is posited as a simpler approach to wide cross-disability accessibility.

Keywords: Cross-cultural product and service design, disability, accessibility, framework, input, output, interaction, user interface, modality independence.

1 Introduction

Information and communication technologies (ICT), and other devices and systems with ICT-like interfaces, are widely used for work, education, entertainment, and daily living. However, many people with disabilities find it difficult or impossible to use these systems for a variety of reasons [1]. This difficulty is frequently a result of a mismatch between the user and the task or between the user and the interface. For many ICT systems, the interface is the only barrier to completing a task for a user with a disability. If the user instead had an interface that fit his or her needs, the same task could be completed successfully.

Designing accessible interfaces can be challenging because many factors must be considered. First, the specific needs, characteristics, and abilities of people with disabilities are very diverse [2]. A second factor is that not all accessibility strategies can be used in all design contexts. For example, accessibility may be provided in some software by using established accessibility application programming interfaces (APIs), which work with a user's assistive technology (AT). However, other systems have *closed functionality*, where particular functionality can only be accessed through the built-in user interface. With closed functionality APIs cannot be utilized, so these systems must provide a built-in, directly accessible interface to the user.

A third factor is the many accessibility standards, regulation, guidelines, and published lists of strategies for ICT [3–5]. Two key accessibility regulations in the United States that a designer needs to understand are the Accessibility Standards of the Americans with Disabilities Act and Section 508 of the Rehabilitation Act of 1973. Many other countries have similar types of accessibility legislation in place or under development. The International Organization for Standardization has a number of standards that relate to accessibility, including ISO 9241-20:2008 (ICT accessibility guidelines), and ISO 9241-171:2008 (software accessibility). The Web Content Accessibility Guidelines (WCAG) 2.0 are accessibility guidelines for web content and applications, but also are being cited as guidance for non-web software and content [6]. In addition to the standards and regulations, there are a number of books, chapters, and articles available on accessibility.

The documents and accessibility standards listed above, and others like them, provide accessibility guidance or regulations, but are complex with many detailed provisions that require time and study to understand. The volume and complexity of the information can be overwhelming to new designers or designers new to accessibility. It is difficult for these designers to prioritize accessibility so that the most important features are addressed first [2] and to identify strategies that can address multiple requirements simultaneously.

To facilitate the incorporation of accessibility into the design process, designers need a simpler accessibility framework. In this paper, we describe a framework of basic modality-independent interaction that can be used to more simply understand and address the needs of people with physical and sensory disabilities ranging from mild to severe for many contexts. In this framework we define modality-independent input and output channels and suitable interface styles. With modality-independent interaction, a user can interact with an interface using any input or output modality that fits his or her needs. The input, output, and interaction may be provided by the system itself or through a user's AT. The model is not meant to replace comprehensive accessibility guidelines, but to provide a simpler way to address basic access.

Modality-independent input and output channels have been considered before and are the basis for many of the current accessibility guidelines. On the output side, the information must be available to AT in modality-independent form or in a form that can be easily converted to a modality-independent form in order to allow AT to present information in different modalities to fit the user [7] (this concept is also reflected in WCAG 2.0 Guideline 1.1 [8]).

On the input side, modality-independent input (or an input method that can easily be translated across modalities) is also required. Guideline 2.1 from WCAG 2.0 [8] requires keyboard input to address this need for web content. In this paper we extend these input and output concepts to more types of systems.

2 Modality-Independent Interaction Framework

In developing the framework of modality-independent interaction, we considered the needs of users without disabilities and the needs of users with sensory or physical disabilities ranging from mild to severe. The most severe sensory disabilities considered were people who are completely unable to see, unable to hear, and unable to both see and hear. The most severe physical disability we considered was where people have control over only a single customized switch.

In order to be accessible, a system must have accessible input, output, and interaction. These can be transformed into a wide range of forms to fit many users' needs.

2.1 System Input

In order to operate an interface, the user must provide input to the system. However, people with physical disabilities may be precluded from using various input devices. To be controllable by a wide range of users, systems must accept modality-independent input, which can be provided by a wide variety of input devices.

We define *modality-independent input* as **non-time-dependent encoded input**. The type of input depends on the capabilities of the system, but must be in a standard format that can be emulated by AT over a standard connection. For computer and mobile-device operating systems, encoded input is keyboard entry using the standard human interface device drivers over USB or Bluetooth. Other examples of encoded input include DTMF tones used by touchtone interactive menu systems and Baudot tones used with text telephones (TTYs). Depending on the interface style, the encoded input might be a subset of that which is accepted by the system. For an interface that requires text input, the encoded input is all of the characters available in that language. For a more limited interface, all interaction may be available using just a few keys (e.g., Shift, Tab, and Enter).

Modality-independent input can be generated through many methods including physical keyboards, Morse code using switches or sip-and-puff [9], chordic keyboards (including Braille keyboards), speech recognition, handwriting recognition, gestures, brain-control interfaces [10], eye or other body movements, or any type of pointing or scanning in combination with a real or virtual keyboard/keyset.

It is important that input not depend on time for several reasons. Some people have difficulty generating specific movements within a particular time span while others need additional time to plan their movements in advance. Generating input with a limited number of switches can take a significant amount of time. Single-switch scanning methods are all time-dependent and customized to the users' abilities. Scanning methods cannot generate input at any arbitrary rate for time-dependent interfaces.

2.2 System Output

In order to perceive the output and feedback from a system, the output needs to be available in a modality that each person can use. Modality-independent output can be transformed or translated into forms that fit the user's needs and abilities.

We define *modality-independent output* as **electronic text**. To be completely accessible, the electronic text output must convey all of the information necessary for full interaction. To work with AT, the system must transmit the text over a standard connection in a standard format. The exact connection and format depends on the system capabilities and content language. Simple ASCII character encoding is sufficient for English text, but other encodings may need to be used for other languages.

Electronic text output can be transformed into a wide range of accessible output in different modalities. Text can be transformed into speech, provided on a braille display, or rendered in a user's preferred font size, color, and typeface. Electronic text can also be translated with varying degrees of success into other languages, including sign language presented by computer-generated avatars [11].

The presentation of system output could be improved by adding semantic or modality-specific markup to the text. Semantic markup describes the underlying meaning of content, which could potentially be provided in different ways for different users. Modality-specific markup defines the presentation of content, for example, the exact pronunciation of a name or the font of text. While additional markup may improve the user experience, plain electronic text is the minimum required for modality-independent accessibility.

2.3 Interface Styles

Accessibility relies not only on accessible input and output channels, but also on a suitable interface style. There are a wide variety of interface styles including direct manipulation, immersive environments, menu selection, form fill-in, command line, and natural language interfaces [12]. However, some interface styles do not translate well into other modalities. For example, in a visual direct manipulation interface the relationships and commands represented through spatial arrangement or pointing gestures are difficult for many people who are blind.

We have identified three general categories of interface styles that are well-suited to modality-independent interaction: command line, single-keystroke command, and linear navigation interfaces. As much as possible, interfaces should be designed so modality-independent input or output can be used independently or together. For example, a person might like to see the interface on a kiosk's full screen (modality-*dependent* output), but interact with modality-independent input using a set of customized switches on his or her wheelchair.

Command Line Interface. With a command line interface, the user submits a string of characters or text to issue commands. Such commands might range from the highly structured format of a command language to natural language input. The acceptable text input depends on the interaction context. The text could be entered using input hardware on the system or through the modality-independent encoded input channel.

Single-Keystroke Command Interface. With a single-keystroke command interface, commands are issued in direct response to a keystroke or combination keystroke. Users do not have to submit the text or command as they do with a command line interface. Many common computer keyboard shortcuts are examples of single-keystroke command interaction (such as pressing Ctrl/Cmd+C to copy). To be accessible, the user must also be able to execute combination keystrokes in a serial manner, such as through Sticky Keys.

Linear Navigation Interface. With a linear navigation interface, users linearly navigate the elements of a screen by moving a highlight or focus cursor. Once on a desired element, they activate or otherwise interact with it. By default, the focus should be navigable to all elements and wrap with notification at screen boundaries.

At a minimum, the encoded input channel must accept two commands that can be provided with a single switch: *Next-element* and *Action* (which on a keyboard might be the Tab and Enter keys, respectively). A third command, *Previous-element*, is strongly recommended because many users can then step back to an element rather than having to wrap. As much as possible, elements should be fully operable with only the *Next-element* and *Action* commands. This allows a switch user to interact with elements without changing to full keyboard mode. Elements with more complex interaction (e.g., text entry) should accept more input from the encoded input channel.

To simultaneously meet the needs of people with different disabilities, there are some navigation compromises. Linear navigation must be provided by default because it allows a person to traverse all elements along a single dimension, like moving down a list. This makes it easier to search non-visually for an element and to navigate with limited switch input. Another compromise is the level of navigation. People need to perceive all elements, so navigation should move to every element (both interactive and non-interactive) by default. This may be an inconvenience for people with physical disabilities (who may want to navigate only to interactive elements) or for more advanced users (who may want to navigate at different hierarchical levels; e.g., navigating by heading and then switching to element-by-element navigation). To enhance the interface, different modes of navigation might be supported, but for modality-independent interaction the requirement is for linear navigation to all elements.

3 Discussion

In the proposed modality-independent interaction framework we outline a simpler and more holistic approach to providing basic accessibility across disabilities than by trying to follow a myriad of accessibility provisions in regulations and guidelines. Designers only need to focus on providing quality modality-independent input, output, and interaction (through appropriate interface styles).

In this section, we discuss the trade-offs associated with different interface styles regarding usability. Additionally, the proposed framework has limitations related to tasks that are inherently inaccessible and the current state of technology. We conclude this section with a discussion of the benefits and uses of the framework.

3.1 Interface Style Considerations

Different interface styles may be suitable for different types of devices, tasks, and contexts of use. To determine the most suitable interface style, a designer must balance an interface's efficiency with the cognitive and memory demands imposed on the user. Designers should consider providing multiple interface styles in parallel so that users can pick that which fits them best. For example, an interface could allow for touch screen use, navigation, and provide direct keyboard shortcuts for efficiency.

Interface Efficiency. An efficient interface can reduce frustration and increase productivity. Efficiency is a function of the interface, the user's abilities, the method of input, and the output provided by the interface.

Input. Input that is most efficient varies widely between users and tasks. While direct single key entry can be very efficient, a large number of choices may be out of the range of motion for a user. Command line input can be efficient for large sets of choices or functions if the user can generate text fairly rapidly, but can be extremely slow if text entry is slow. Navigation might be faster for people who have slow text entry but slower for someone who can enter text quickly. Input efficiency is only one aspect of overall interface efficiency. Users must also receive output from the system, which may be more or less verbose.

Output. Interfaces with more electronic text output are less efficient to use because users must take more time to read or listen to the output. The amount and type of output to be provided depends on both the anticipated users and the interface style.

With both command line and single-keystroke command interfaces, text output may range from menus of commands to a simple prompt. A prompt may be efficient for an expert user who has memorized many commands or for systems that process natural language input. However, many users of such interfaces require more than prompts. They may need an explicit menu of the available options (for example, "To make a payment, press 1. For your balance, press 2….") or a method to get help, documentation, and instructions at any time.

For a linear navigation interface, the output is a text representation of the element that is in focus, giving its purpose in the interface. Such text should include important information necessary for operation; for example, the element's type or role, its current status, how it relates to other elements in proximity, etc. Since one is only receiving system output for a single element at a time (for example, "Payment, button" or "Balance, button"), this can be more efficient than systems that present menus from which a user must choose.

Cognitive and Memory Demands. Interfaces place cognitive and memory demands on users, some of which may make a particular interface style unsuitable. For example, interfaces that require significant memorization of commands or keystrokes are unsuitable for novice users and for public kiosk implementations.

Linear navigation places relatively low memory demands on users because users navigate lists of options. When on an element, the user only needs to decide yes or no and then activate it or move to another one. However, navigating and then selecting may be more cognitively difficult than directly selecting a desired element [13].

Command line and single-keystroke command interfaces allow a user to directly select a choice, but people must know the desired command(s). This involves memorizing commands, reviewing help documentation, or carefully listening to or reading a menu of options and associated commands. Single-keystroke commands that seem to be arbitrarily assigned may be more difficult to remember than command line input that uses familiar words or phrases. Natural language interfaces may be easier to use because they do not require as much learning and memory as other more structured command formats.

3.2 Task-Related Limitations

There are limitations to the proposed framework that are inherent to tasks. In these cases, changing the interface would be insufficient for accessibility; the task itself would need to be fundamentally changed to make it accessible across disabilities.

It is important to note that tasks with such limitations are not automatically inaccessible to all people with disabilities. Instead, such aspects represent barriers to universal accessibility for some people who have physical or sensory disabilities.

Time-Dependent. Tasks that depend on time can be difficult or impossible for people who interact with or perceive information from systems slowly. It might take the person extra time to move, manipulate, or control an interface. Some people may take more time to navigate and find the option they want. Getting information in some modalities is inherently slower than looking at a screen and reading text.

Examples of time-dependent tasks are real-time events such as participating in an auction or operating a motor vehicle. Other tasks may have prescribed time limits, such as taking a standardized test. Multimedia is inherently time-dependent. Finally, many tasks are implicitly time-dependent because people must meet some level of productivity and complete tasks in an efficient, timely manner.

Complex Path-Dependent. It can be difficult for people with sensory and physical disabilities to create, trace, or follow a path. A path often takes place in spatial dimensions but may have other dimensions as well (e.g., continuous pressure). While it might be technically possible for a person to define a set of points along a path, such input is inefficient and may be inaccessible for practical reasons, especially with long, complex paths. Some people have difficulty with path input because they cannot see the path they are creating or other paths that may already be displayed.

Some tasks require complex path input, such as using a drawing application. Many tasks that require manipulation of a device or interface are path-dependent, for example aiming a camera to scan a barcode requires manipulation in three dimensions. Some path-dependent tasks can be fundamentally changed so they do not require path input. For example, the act of driving a vehicle is a time- and path-dependent task, whereas entering a destination into an autonomous vehicle is a text entry task.

Simultaneity-Dependent. Tasks that depend on simultaneous perception and/or operation can be difficult for some people with disabilities. People who must use a single body part or implement to activate all controls would have difficulty with any simultaneous input. Because of limits to attention, simultaneous output can be difficult for everyone to monitor, follow, or track. People with sensory disabilities may have additional problems because of limitations of the output modality they need to use.

Some tasks require a person to be able to perform simultaneous actions. Playing a piano involves pressing keys simultaneously to play chords. Other tasks require a person to perceive simultaneous streams of information. For example, it is difficult to listen simultaneously to a teleconference and a screen reader reading a related document. Reading subtitles or captions might be difficult if the text is replaced quickly and one is also trying to follow fast action or visually displayed information.

Experience-Dependent. An input or output experience may be so much a part of a task that changing the experience would be a fundamental change to the task. Users who cannot perform the particular input required, or perceive the output, will be precluded from the experience and the task. There may be ways of providing equivalent functionality and access, but that is not the same as providing the same experience.

Some tasks require a person to be able to perform specific actions. For example, games may have gesture-based controls where making the gesture is part of the gaming experience. Changing the input would fundamentally change the game. Biometric security devices require a person to scan a body part that he or she might not have or might not be able to present to the scanner.

Additionally, some tasks require a person to be able to perceive information in a particular modality. For example the full experience of viewing artwork, watching a movie, or listening to music cannot be simply replicated in a different modality or in text. There are many types of statistical graphics and data visualizations that require visual pattern recognition in order to quickly see trends or patterns that might not be apparent from the raw data. While the meaning or interpretation of some simple data visualizations can be presented through text, there is not always a clear way to provide more complex data visualizations in other modalities. If the modality of the information presented is important to its meaning, then the information is experience dependent.

3.3 Current State of Technology

In the current state of technology, there is not yet a good, single protocol for AT interconnection. Current input hardware AT typically uses standard human interface device drivers on serial or USB ports. Output to hardware AT relies on specific device drivers and standard system connections. If AT supported a standard electronic text format, device drivers would not be necessary. In order to have cross-disability access to people who can bring their own AT, standards would need to be adopted or developed. Even without a universal AT connection standard, the modality-independent interaction framework is still useful today.

3.4 Benefits and Uses of the Framework

The modality-independent interaction framework can be used for a number of purposes. Those developing new accessibility APIs or accessible systems can use it to ensure that people with severe disabilities have sufficient interaction (for example, cross-disability navigation interfaces need a concept of element focus). The framework is particularly useful for designers who need to provide built-in accessibility. Those designing devices that are traditionally closed to AT could potentially make systems fully accessible to people with sensory and physical disabilities through only a simple connection. Even on completely closed devices where AT connections are not allowed, the framework could help designers focus on the important characteristics of the interface, such as what output text is important and what limited interaction must be supported. For example, the important text output could be provided in speech output for people who are blind, and navigation could use a simple button set that could be used by many people with physical disabilities.

With the framework, designers only need to design one cross-disability accessible user interface for a very wide range of disabilities. It is true that such cross-disability access is not optimized for any particular user or disability category, but providing a set of disability-optimized interfaces would take significant resources for development and maintenance. Some of the compromises inherent in a cross-disability interface could be eliminated through user profiles or settings. Moreover, considerations about efficiency or optimization may not be as important for simple public devices, like a check-in kiosk.

4 Conclusion

In the future, a person might come up to a kiosk in a wheelchair with his AT software running on a tablet. Using near field communication, he could establish a wireless connection to the kiosk. After connecting, he could then look at the kiosk's screen and use the joystick on his wheelchair to navigate the screens and make selections. Another user, who is blind, could go up to the same kiosk and pair her smart phone with the kiosk. She might use swiping gestures on her smart phone to navigate the kiosk's modality-independent interface. For each item on the screen, the kiosk would transmit text to her smart phone, which would speak to her privately through headphones.

For tasks that do not have inherent barriers to accessibility, the proposed framework simplifies accessible interface design and facilitates a wide range of accessibility. Additionally, the framework helps designers avoid having to create many separate modality-specific interfaces to meet diverse user needs.

Acknowledgement. The contents of this paper are based on work carried out with funding from the National Institute on Disability and Rehabilitation Research, U.S. Department of Education, grant number H133E080022 (RERC on Universal Interface and Information Technology Access). However, the contents do not necessarily represent the policy nor imply endorsement by the funding agencies.

References

1. Vanderheiden, G.C., Jordan, J.B.: Design for people with functional limitations. In: Salvendy, G. (ed.) Handbook of Human Factors, 4th edn., pp. 1409–1441. John Wiley & Sons, Hoboken (2012)
2. Vanderheiden, G.C.: Accessible and Usable Design of Information and Communication Technologies. In: Stephanidis, C. (ed.) The Universal Access Handbook, pp. 3-1–3-26. CRC Press, Boca Raton (2009)
3. Hodgkinson, R.: 10th Report on International ICT Accessibility Standards: Proposed, Being Developed and Recently Published (2009), http://tiresias.org/research/standards/report_10.htm
4. International Organization for Standardization/International Electrochemical Commission (ISO/IEC): ISO/IEC TR 29138-2:2009: Information Technology—Accessibility Considerations for People with Disabilities—Part 2: Standards Inventory. ISO, Geneva (2009)
5. Vanderheiden, G. C.: Standards and Guidelines. In Stephanidis, C. (ed.) The Universal Access Handbook, pp. 54-1– 54-21. CRC Press, Boca Raton (2009)
6. Architectural and Transportation Barriers Compliance Board: Telecommunications Act Accessibility Guidelines; Electronic and Information Technology Accessibility Standards. Federal Register 76(236) pp. 76640–76646 (December 8, 2011)
7. Vanderheiden, G.C., Henry, S.L.: Everyone Interfaces. In: Stephanidis, C. (ed.) User Interfaces for All, pp. 115–133. Lawrence Erlbaum, Mahwah (2001)
8. World Wide Web Consortium (W3C): Web Content Accessibility Guidelines (WCAG) 2.0. W3C Recommendation (2008), http://www.w3.org/TR/WCAG20/
9. Fleming, B., Lin, A., Philips, B., Caves, K., Cotts, M.: Morse code demystified: A powerful alternative for access to AAC and computers. Paper Presented at CSUN 2003 (2003), http://www.csun.edu/cod/conf/2003/proceedings/71.htm
10. Wolpaw, J.R., Birbaumer, N., McFarland, D.J., Pfurtscheller, G., Vaughan, T.M.: Brain–computer interfaces for communication and control. Clin. Neurophysiol. 113, 767–791 (2002)
11. Huenerfauth, M., Hanson, V.L.: Sign Language in the Interface: Access for Deaf Signers. In: Stephanidis, C. (ed.) The Universal Access Handbook, pp. 38-1 – 38-18. CRC Press, Boca Raton (2009)
12. Shneiderman, B., Plaisant, C.: Designing the User Interface: Strategies for Effective Human-Computer Interaction, 5th edn. Addison-Wesley, Boston (2010)
13. Dowden, P., Cook, A.M.: Choosing effective selection techniques for beginning communicators. In: Reichle, J., Beukelman, D.R., Light, J.C. (eds.) Exemplary Practices for Beginning Communicators, pp. 395–429. Brookes, Baltimore (2002)