Evaluating the Accessibility of Digital Audio Workstations for Blind or Visually Impaired People

Gemma Pedrini, Luca Andrea Ludovico[a] and Giorgio Presti[b]

Laboratorio di Informatica Musicale, Dipartimento di Informatica "Giovanni Degli Antoni",
Università degli Studi di Milano, Via G. Celoria 18, Milano, Italy

Keywords: Accessibility, Blind or Visually Impaired (BVI), Digital Audio Workstations.

Abstract: This paper proposes a methodology to assess the accessibility for blind or visually impaired people of music production software known as Digital Audio Workstations. The products chosen for the tests are Cockos REAPER, Avid Pro Tools, and Steinberg Cubase, three of the most popular solutions falling in this category. Both Microsoft Windows and macOS versions were tested, since these two operating systems natively integrate assistive technologies which provide a further layer to be considered. The degree of accessibility was evaluated in relation to the possibility for blind or visually impaired people to invoke key functions and perform basic operations. Finally, a focus group with visually impaired professional music producers was organized in order to assess the proposed evaluation methodology.

1 INTRODUCTION

Accessibility means enabling as many people as possible to use a resource, even when those people's abilities are limited in some way. The *International Classification of Functioning, Disability and Health* (ICF) denotes with the term *disability* the negative aspects of the interaction between an individual and that individual's contextual factors. A first assumption is that an environment can be defined as *accessible* when an individual with any impairment can "function independently". Another assumption is that there is some level of function that can be called *minimally acceptable* (Steinfeld et al., 1999).

The *UN Convention on the Rights of Persons with Disabilities* (CRPD) (United Nations, 2006) at Article 9 – "Accessibility" encourages appropriate forms of assistance and support to persons with disabilities so as to ensure their access to information, promotes their access to new information and communications technologies and systems, and fosters the design, development, production and distribution of accessible information and communications technologies and systems at an early stage, so that these technologies and systems become accessible at minimum cost.

---
[a] https://orcid.org/0000-0002-8251-2231
[b] https://orcid.org/0000-0001-7643-9915

The theme of accessibility in the context of Information Society, also known as *e-accessibility* (Klironomos et al., 2006), concerns the integration of all users into the Information Society, including people with disabilities. Such a subject is tightly connected to *e-inclusion*, which aims to prevent the risk that people with lack of digital literacy, poor access to technology and some form of impairment are left behind, thus experiencing digital exclusion.

Another term often used in parallel with accessibility is *usability*, an adjective synonymous with fit to use, functioning, operational, serviceable, valid, and working. Usability concerns the fulfillment of functional requirements, which makes it different from accessibility. Nevertheless, some scholars use the two expressions in parallel, stating that they both are usually defined in terms of observed task performance, and together represent the concept of person-environment fit (Steinfeld and Danford, 1999). A comprehensive review about accessibility, usability and universal design can be found in (Iwarsson and Ståhl, 2003).

In the wide context of accessibility, computer applications presents very specific requirements (Kavcic, 2005). As stated in (Mozilla, 2019), most software tools assume that users can easily perform all of the following tasks: read and react to text and images displayed on the screen, type on a standard keyboard, select text, pictures, and other information using a

mouse, react to sounds being played. Conversely, people with special needs can experience problems in performing one or more actions, which prevents them from using, partially or completely, even popular computer applications.

Focusing on visual impairments, this category includes the range from low vision (low-grade difficulty in the use of a visual display) to full blindness (no possibility to enjoy graphical content). Please note that, even if the hardest tasks for blind or visually impaired (BVI) people concern the information displayed on the screen, especially graphics and pictorial content, also the use of a pointing device, requiring eye-hand coordination, can pose a problem.

This work focuses on accessibility issues of Digital Audio Workstations (DAWs) for BVI users. In origin, a DAW was a computer equipped with a sound card and ad-hoc software for creating, editing and processing recording-studio quality digital sound (Leider, 2004). A computer-based DAW presents four basic components: a computer, either a sound card or an audio interface, digital audio editor software, and at least one input device for adding or modifying data. Nowadays, DAW is a common definition for a software system that provides the interface and functionality for audio editing and uses a PC as a host for sound generation.

There is also a business interest towards music software accessibility. As stated in (W3C Web Accessibility Initiative (WAI), 2019), accessible design improves overall user experience and satisfaction, especially in a variety of situations, across different devices, and for impaired as well as older users. Supporting accessibility can enhance a brand, drive innovation, and extend the market. Among BVI people, there are users interested in music generation, editing and production, and they often rely on hardware aid devices due to the poor support offered by software tools.

This work aims to shed light on the problem of DAWs accessibility for BVI people, focusing on pure software aspects and leaving out proprietary or MIDI controllers that could help solving the problem. Anyway, the final goal is not to provide an evaluation of accessibility for software DAWs currently available on the marketplace, since new versions of the applications and the operating systems are expected to introduce novel functionalities to support visual impairments; in this sense, the results of our research would soon become obsolete. Rather, we will start from the evaluation of existing software tools in order to propose an objective methodology to rate the accessibility of past, current and future DAW releases. Our proposal has been submitted to visually impaired

professional music producers for appraisal, and their remarks will be reported and discussed.

The rest of the paper is organized as follows: Section 2 will describe the main assistive technologies that help BVI people in using computer devices, Section 3 will focus on the accessibility of DAWs, describing the test protocol and showing the scores obtained, Section 4 will discuss the comments made by experts to the proposed methodology, and, finally, Section 5 will draw conclusions.

2 ASSISTIVE TECHNOLOGIES FOR BVI PEOPLE

According to (U.S. Government, 1998), the adjective *assistive* is assigned to technology designed to be utilized in an assistive-technology device or an assistive-technology service. Concerning the former aspect, the expression *assistive-technology device* identifies any item, piece of equipment, or product system, whether acquired commercially, modified, or customized, that is used to increase, maintain, or improve functional capabilities of individuals with disabilities. Regarding the latter aspect, the expression *assistive-technology service* means any service that directly assists an individual with a disability in the selection, acquisition, or use of an assistive-technology device.

Assistive technologies for BVI people have been widely discussed in literature, e.g. in (Hersh and Johnson, 2010), (Pawluk et al., 2015) and (Shinohara, 2006), to cite but a few. Alternatives to the use of sight in computer interaction typically involve the use of other sensory canals, mainly hearing and touch. To this end, a number of computer-based approaches and aiding tools have been designed and developed.

Screen readers are a category of software aid tools for people who do not have useful vision to read text on the screen. They can analyze, filter, and interpret the content of a computer display and reproduce it as audio output through text-to-speech synthesis (see Figure 1) or pilot a refreshable Braille display (see Figure 2). Since the beginning of the '90s, the interface of most operating systems has been no longer exclusively textual, but it has adopted graphical components and pictograms to convey information, thus becoming a so-called *graphical user interface* (GUI). The software running under graphical operating systems typically presents the same characteristics, supporting non-textual information and relying on a number of graphical controls. For this reason, recent screen readers must be able to convert into an alternative representation not only text, but also graphical elements; in this sense, associated metadata, tags,
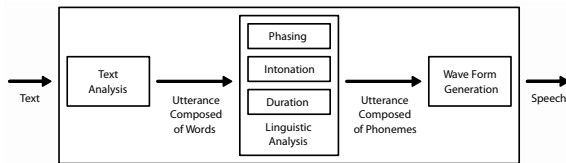
Figure 1: Block diagram of a typical text-to-speech system.

Figure 2: A 40-cells refreshable Braille display. Image by Sebastien.delorme, CC BY-SA 3.0.

and automatic recognition algorithms can help. Many operating systems integrate their own screen readers and speech synthesizers: e.g., *Orca* for Linux, *Narrator* for Microsoft Windows family, and *VoiceOver* for Apple macOS.

It is worth mentioning another category of aid tools specifically addressing visually impaired users: *screen magnifiers*, which are capable of presenting the output on a larger scale. The enlarged portion of the screen, called the *focus*, includes the content of interest and an improved representation of the pointer or cursor. The focus follows pointer movements, and usually it can be invoked through a shortcut and adjusted in size depending on the user's needs. Modern operating systems integrate screen magnifiers as well.

Even if designed for different purposes, assistive technologies of interest in this context can also include specific *hardware controllers*, provided that they support bi-directional communication from/to the DAW. Examples include motorized faders and knobs. Anyway, in the analysis of DAWs accessibility presented in Section 3 we will deliberately ignore the role played by additional hardware components, due to their high number and variety, and, above all, to their extraneity in relation to the original software tools.

When designing software interfaces, developers should take into consideration the compatibility of their products with assistive technologies (Edyburn, 2004). Common problems of accessibility in software include: controls identified only by images, without textual tags; absence of explanatory alternative texts; wrong tag-reading priority, with descriptors not following a logical/functional order; the need to use the mouse to reach controls that cannot be selected through speech-synthesis focus; display-related information not easily reachable or completely unreachable by keyboard navigation.

There is a number of good practices that can be followed to improve software usability for BVI people. A fundamental rule is to provide titles and labels for each element of the interface. The meaning of icons and other pictorial indications should be conveyed by alternative text, too. The activation/deactivation of controls like buttons should not be rendered only visually, but also by providing auditory feedback to users of assistive technologies. Overlay notifications like tooltips should persist for a time sufficient to be read through the screen reader, or their timing should be adjustable. In general, too complex graphical layouts should be avoided, since they pose serious problems of accessibility and often require mouse interaction. Even if the default layout has not been designed for BVI people, a clear, intuitive and highly-customizable interface, accompanied by accessible documentation, could help. For example, the support of high contrast color combinations and the possibility of switching foreground and background colors are useful features for BVI users. All interface elements should be reachable through speech-synthesis focus by using appropriate labels. Moreover, programmers should rely on the APIs provided by operating systems for standard controls and follow the guidelines for accessibility in the design of customized ones. Finally, software should be tested through ad-hoc tools to ensure the accessibility via keyboard or mouse of interface components such as modules, menus, selection curtains, combo boxes, check boxes, etc.

In conclusion, assistive technologies can be employed to provide context information, command descriptions and feedback. Unfortunately, in sound-oriented software these purposes can pose some problems: on one side, an excessive stimulation of hearing can cause cognitive overload in BVI users; on the other side, it is hard to convey some types of information (e.g., waveforms or automation patterns) via a Braille display.

3 ACCESSIBILITY OF DAWs

In order to evaluate the accessibility of DAWs, we analyzed the reachability for a BVI user of a set of common functions and operarations. To this end, we selected and tested three very popular software solutions:

1. Cockos REAPER,[1] version 5.95 for Windows and version 5.921 for macOS;

2. Avid Pro Tools,[2] version 2018 7.0 for Windows and 2018.10 for macOS;

3. Steinberg Cubase Pro,[3] version 9.5 for Windows and macOS.

Tests were performed under Windows 10 version 1803 (build SO 17134.472) and macOS Mojave 10.14.

It is worth underlining that, at the moment of finalizing this paper (July 2020), such versions have been already surpassed by more recent releases, introducing new features in the graphical interface and, more in general, presenting some differences concerning accessibility. Anyway, we recall that the research question we want to address in this paper is not to determine which DAW is currently the most accessible one for BVI people, but rather to highlight which aspects programmers should take into account when designing a DAW, and ultimately to propose a general methodology for an objective evaluation of DAWs' accessibility. From this perspective, the results achieved by specific software versions are not relevant.

3.1 Assessment Metrics

The metrics we propose takes into account both the usability and the access speed for the analyzed functions. To this end, we define an ad-hoc scale with values ranging from 0 (minimum) to 4 (maximum). These values correspond to the following accessibility levels:

- *Level 0.* The command under exam cannot be invoked via keyboard, and the vocal synthesis does not recognize the presence of the control in case of interaction through the mouse;

- *Level 1.* The command can be reached from the keyboard using the cursor or via a shortcut only in a specific context, but context information is not clearly communicated to the user, e.g. via speech synthesis. Moreover, there is no feedback about the success or failure of the invoked operation, concerning both the activation of the command and its effects;

- *Level 2.* The command can be invoked through the keyboard, using the cursor or via a shortcut. Speech synthesis does not provide a feedback, but

it is possible to check the effectiveness of the command;

- *Level 3.* The command can be reached from the keyboard using the cursor or via a shortcut, and voice synthesis produces a feedback, which lets the user check the effectiveness of the command;

- *Level 4.* The command is fully accessible from the keyboard and perfectly integrated within the graphic environment.

3.2 Choosing the Screen Reader

The choice of the screen reader is critical for the success or failure of many operations. For both the operating systems under exam we referred to the guidelines suggested by the *American Foundation for the Blind*.[4] In order to select the most suitable speech system, this institution poses a number of questions: What version of the operating system will be used? Is the screen reader compatible with such a version? Are there known system configurations with which the screen reader does not work (color schemes, common video cards, etc.)? What synthesizers are, or are not, supported? Considering the most common applications, are there known limits with the screen reader? How much speech does the screen reader automatically add during standard functions, such as selecting or scrolling items? Can the amount of speech be adjusted to suit the user's skill level and preferences? How difficult is to change simple standard features (e.g., the voice rate)? Is the manual accessible and accurate? Is there a tutorial in a usable format?

After carefully evaluating the mentioned questions, we selected *NVDA* as the screen reader to be used under Windows. *NVDA*, standing for NonVisual Desktop Access, is an open-source screen reader that uses the *eSpeak* speech synthesizer and *SAPI 4* and *SAPI 5* synthesizers.[5] It is free and updated frequently, about every 3 months, thus keeping the pace with technological innovations. This choice has proved to be effective especially in the program testing phase, allowing an easy investigation of the screen through the mouse. *NVDA* is able to read the elements currently positioned under the pointer. Speech synthesis does not interfere with common mouse operations, such as button clicking or wheel scrolling. During our tests, we have experienced only two problems: sudden and apparently unmotivated slowdowns in the reactivity of invoked operations and unexpected restarts. In the latter case, speech synthesis stopped

---

[1] https://www.reaper.fm/

[2] https://www.avid.com/pro-tools

[3] https://new.steinberg.net/cubase/

[4] https://www.afb.org/

[5] https://github.com/nvaccess/nvda

working, and the proper behavior could be resumed only by restarting the program in use.

Concerning macOS, Apple devices embed a proprietary voice synthesis called *VoiceOver*. Installed by default, perfectly integrated in the operating system, constantly updated and guaranteed to be compatible with all applications, this solution provides excellent performances with all Apple devices.

3.3 Test Protocol

The analysis consisted in a test conducted by a visually impaired user, expert in the field, on a significant set of DAW commands. The tester invoked the most common operations of a recording studio, that can be gouped into the following categories:

- *Transport*. Basic and advanced playback controls;

- *Track*. Operations on sequencer tracks concerning navigation, management and controls;

- *Editing*. Commands to cut, copy, paste or duplicate events, time representations of events and commands to change the content of each track;

- *Mix*. Audio stream production and post-production processing;

- *Project*. File and configuration management.

Each command was tested on any version of the DAWs under exam, and an accessibility level was assigned according to the scale described in Section 3.1. All combinations to achieve the same result were tested (menu, keyboard, and mouse commands), and scores reflected the best option, namely the most accessible method. Final results are shown in Table 1, which provides a detailed comparison between the 3 products in their Windows and macOS versions.

Let us briefly discuss the outcome of our analysis. REAPER emerges as the most accessible DAW, scoring very well under both macOS and Windows. Only a few operations among the commands under exam received a low grade. In order to understand such an impressive result, we have to mention that Cockos REAPER 5.9 has already been analyzed and adapted by a community of BVI people, a working group known as *Reaper Accessibility*. Their activities resulted in the design and implementation of a dedicated plugin called *OSARA* (Open Source Accessibility for the REAPER Application), [6] an extension that makes the DAW accessible to screen-reader users by interfacing directly with the speech synthesizer. Even if *OSARA* is the result of an independent community

of users, it is worth remarking the involvement of the BVI community in the product design.

Conversely, Cubase 9.5 received a poor score, showing very little attention towards accessibility issues for BVI people. Nevertheless, Cubase Pro 10.5 (the most recent version available on the marketplace at the moment of writing) has substantially solved the problems that affected the previous release.

The test phase, conducted by a low-vision user, made some issues typically unknown to standard users emerge. First, many operations present different effects depending on the context in which they were performed (e.g., editing commands), hence the importance to clearly identify the position of the focus. This aspect can be fundamental in the evaluation of accessibility, above all when the user has no residual sight.

Another problem concerns the possibility for BVI people, like our test user, to access documentation. Even if all the DAWs under exam have a complete user's guide, sometimes accessing it through assistive technologies was a hard task. Therefore, it was essential to retrieve supplementary material, provided by developer communities such as Github.com or shared on platforms such as Youtube.com. Focusing on the latter approach, video tutorials often use the voice to explain instructions shown via desktop recording, therefore BVI users cannot understand the context. Furthermore, instructions are invoked via both keyboard and mouse inputs, and the sequence is not easily repeatable through assistive technologies.[7] Concerning audiovisual material made accessible to the blind, it is worth mentioning the *Access Music Tech* channel,[8] a community-driven virtual place dealing with assistive music technologies. The contents of this channel have been fundamental in finding DAW instructions for macOS software, while the corresponding instructions under Windows were inferred by attempts, also considering the mapping of keyboard commands under the two operating systems.

4 ASSESSING THE PROTOCOL

In order to check the validity and significance of the data gathered by our protocol when applied to the mentioned DAWs, we organized a focus group involving BVI experts in the field of music production and asked them to provide observations and remarks about the scores obtained by the test.

---

[6]https://osara.reaperaccessibility.com/

[7]Anyway, desktop recording can help visually impaired people with residual sight.

[8]https://www.youtube.com/channel/UC-rL46RtrPdb_I3awd9eJow

Table 1: Scores in the range 0 (minimum) to 4 (maximum) obtained by the DAWs under exam for each tested task.

Task	Feature	REAPER		Pro Tools		Cubase	
		OSX	Win	OSX	Win	OSX	Win
Transport	Start	4	2	4	2	0	0
	Stop	4	2	4	2	0	0
	Pause	4	2	4	0	0	0
	Record	4	3	4	2	0	0
	FFW/RW	4	1	4	2	0	0
	Home/End	4	4	3	2	0	0
	Loop on/off	4	4	4	3	0	0
	Cursor snap/scrub	4	4	0	0	0	0
	Cursor fine locating	0	4	0	0	0	0
	Create marker	4	3	4	3	0	0
	Marker window	3	3	0	0	0	0
	Move to marker	4	3	0	0	0	0
	Delete marker	3	4	2	2	0	0
	Grid settings	4	2	0	0	0	0
	BPM	4	4	2	4	0	0
	Signature	4	4	2	2	0	0
	Tempo track	0	0	0	0	0	0
	Metronome on/off	4	4	3	4	0	0
	Metronome setup	3	4	4	0	0	0
Track	Create track	4	4	4	4	0	0
	Rename track	4	4	3	4	0	0
	Delete track	4	4	4	4	0	0
	Track focus	4	3	3	1	0	0
	Track selection	4	4	4	1	0	0
	Track type	0	0	4	3	0	0
	I/O Mapping	0	0	4	2	0	0
	Arming track	4	3	4	4	0	0
	Record mode	4	4	2	3	0	0
Editing	Import audio	3	4	4	4	0	0
	Sample selection	4	4	0	0	0	0
	Select all	2	4	4	4	0	0
	Ripple Edit	4	4	0	0	0	0
	Cut	4	4	4	1	0	0
	Copy	4	4	4	1	0	0
	Paste	4	4	4	1	0	0
	Duplicate	4	4	4	1	0	0
	Split	0	4	0	0	0	0
	Undo/Redo	4	3	4	4	0	0
Mix	Show mixer	3	0	4	0	0	0
	Show master track	4	4	4	3	0	0
	Track fader	4	0	4	0	0	0
	Master fader	3	3	4	3	0	0
	Track pan	3	2	4	3	0	0
	Mute/Solo	4	3	4	4	0	0
	Insert FX	4	3	2	0	0	0
Project	Open project	4	4	4	2	1	1
	New project	4	4	4	4	1	1
	Save project	4	4	4	4	1	1
	Save project as	4	4	4	4	4	4
	Close project	4	4	4	4	1	1
	Close all	4	4	4	4	1	1
	Mixdown	4	4	3	4	0	0

We received answers from two experts who have been working for years with DAWs at a professional level.

A first aspect we asked to validate was the set of commands chosen for our tests. In order to be present in all the tested DAWs, we limited it to the basic operations typical of a recording studio, not taking into account articulated sequences of commands, complex sound-processing operations, nor third-party plugins usage. From this point of view, both experts consid-

Table 2: Comparative analysis of the three DAWs under exam.

	REAPER		Pro Tools		Cubase	
	macOS	Windows 10	macOS	Windows 10	macOS	Windows 10
Transport	4	3	3	2	0	0
Track	4	4	4	3	0	0
Editing	4	4	4	1	0	0
Mix	4	3	4	3	0	0
Project	4	4	4	4	1	1

ered the list of commands in Table 1 as exhaustive and significant to test software accessibility.

Moreover, they both agreed on the general outcomes of our investigation, confirming that Cockos REAPER is more user-friendly for a BVI user than other software tools.

As we might expect, they also gave us useful suggestions to refine the test protocol. First, they underlined how the choice of the screen reader may alter results significantly. For instance, one of them suggested a commercial software solution that, in his/her opinion, was essential to cope with REAPER 3rd party plugins under macOS. Consequently, it is important to periodically evaluate the advancements in screen readers and to choose the most suitable one, as discussed in Section 3.2.

Another aspect to consider is the inverse correlation between the richness in the graphical interface and the usability of software by BVI people. In fact, not only ancillary pictorial content is partially or completely useless for this category of users, but it is hard to be rendered through a different sensory canal and can be even misleading with respect to fundamental audio-related information. Consequently, one of the experts suggested to take into account the graphical complexity of the interface and the customization possibilities offered by the DAWs, in order to help some screen readers.

As a final remark, it is worth underlining that professional users often adopt only one operating system and become skilled with a specific product. Thus, their opinions can be relevant when comparing the updates of the same DAW across major or minor revisions, or at least when cross-checking the functionalities of different DAWs under the same operating system, but they rarely have a synoptic view like the one we provided in our analysis, involving many software tools and their variants under different operating systems.

5 CONCLUSIONS

The experience conducted with three popular software DAWs underlined that accessibility for BVI people is an aspect sometimes overlooked by designers and developers.

For the sake of clarity, we have summarized the results obtained by applying our protocol in Table 2, where we have reported the median score of each command category for each software. The table shows that Cubase is inaccessible in almost all its functions, probably due to the use of a proprietary GUI environment instead of the one provided by the operating system's APIs; moreover, the focus of commands is not clear. REAPER 5.95 and Pro Tools 2018 perform better, but the operating system in use deeply influence the results. In particular, macOS provides a higher level of accessibility. During our tests, Windows caused also a number of system crashes until the RAM of the PC was upgraded from 8 to 16 GB, as the simultaneous use of speech synthesis and a DAW takes many resources. In general, we noticed several compatibility issues between such software tools and voice synthesis, which required settings update, additional external devices (that sometimes turned to be incompatible with the software), and program restarts. It will be interesting to apply our protocol to more recent DAW releases, under new versions of the operating systems and with more powerful hardware configurations.

In order to overcome usability issues, a user with special needs has to choose the right combination of products (operating system and DAW), master their settings, run the DAW and speech synthesis on a high-performance computer, and sometimes rely on additional technologies to gain access even to basic functions. Our hope is that impaired users' needs will be given more attention in future software releases.

Concerning future work on the proposed evaluation protocol, the first step is to repeat the analysis with more testers, in order to improve the validation by assessing the statistical validity of the proposed methodology. Then, the obtained results will be the

basis to provide design guidelines for developers.

We also plan to extend tested functionalities to common combinations of operations, such as complete editing tasks on waveforms (e.g., complex editing for tempo correction) or the application of standard behaviors (e.g., select and fade out multiple audio events). Moreover, we need to better clarify the goal, choosing between the evaluation of the accessibility characteristics of a software *per se*, or its usability when benefiting from "external" support technologies, such as suitable extensions, screen readers, operating-system aids, etc. Another aspect to consider in the overall evaluation will be the influence of assistive technologies on the performances of the DAW.

REFERENCES

Edyburn, D. L. (2004). Rethinking assistive technology. *Special Education Technology Practice*, 5(4):16–23.

Hersh, M. and Johnson, M. A. (2010). *Assistive technology for visually impaired and blind people*. Springer Science & Business Media.

Iwarsson, S. and Ståhl, A. (2003). Accessibility, usability and universal design—positioning and definition of concepts describing person-environment relationships. *Disability and rehabilitation*, 25(2):57–66.

Kavcic, A. (2005). Software accessibility: Recommendations and guidelines. In *EUROCON 2005-The International Conference on" Computer as a Tool"*, volume 2, pages 1024–1027. IEEE.

Klironomos, I., Antona, M., Basdekis, I., Stephanidis, C., et al. (2006). White paper: promoting design for all and e-accessibility in europe. *Universal Access in the Information Society*, 5(1):105–119.

Leider, C. N. (2004). *Digital audio workstation*. McGraw-Hill, Inc.

Mozilla (2019). Software accessibility: Where are we today? https://developer.mozilla.org/en-US/docs/Web/Accessibility. Accessed: 2020-01-05.

Pawluk, D. T., Adams, R. J., and Kitada, R. (2015). Designing haptic assistive technology for individuals who are blind or visually impaired. *IEEE transactions on haptics*, 8(3):258–278.

Shinohara, K. (2006). Designing assistive technology for blind users. In *Proceedings of the 8th International ACM SIGACCESS Conference on Computers and Accessibility*, pages 293–294. ACM.

Steinfeld, E. and Danford, G. S. (1999). Theory as a basis for research on enabling environments. In *Enabling Environments*, pages 11–33. Springer.

Steinfeld, E., Danford, G. S., and Danford, G. (1999). *Enabling environments: Measuring the impact of environment on disability and rehabilitation*. Springer Science & Business Media.

United Nations (2006). Convention on the rights of persons with disabilities (CRPD). https://www.un.org/development/desa/disabilities/convention-on-the-rights-of-persons-with-disabilities.html. Accessed: 2020-01-05.

U.S. Government (1998). *S.2432 - Assistive Technology Act of 1998, 105th Congress Public Law 394*. U.S. Government Printing Office.

W3C Web Accessibility Initiative (WAI) (2019). Introduction to web accessibility. https://www.w3.org/WAI/fundamentals/accessibility-intro/. Accessed: 2020-01-14.