ATHEN Report on the Accessibility of Google Documents

October 18, 2011

Contents

Executive Summary

Google Inc. is in the process of improving its Google Application suite to make it more accessible for people with disabilities. To date, many improvements have been made, but the efforts have largely targeted screen reader users and not considered people with other types of disabilities. In order to begin to assess the accessibility of the application suite for a variety of disability types, the Access Technology Higher Education Network (ATHEN) performed functional tests (described in detail below) on two applications: the Google Docs Document List (hereafter referred to as "Document List") and the Documents portion of Google Docs (hereafter referred to as "Documents"). These tests show that many people with disabilities are currently unable to successfully use these applications. No assistive technology tested was able to fully perform every function within these applications, and the level of support for assistive technologies ranged from being able to perform many, but not all of the functions to not being able to use the applications at all. Some of the major problems include:

- speech recognition software users cannot dictate text into, or interact with the application
- keyboard-only users often cannot access the application menu, and thus, much of the functionality of the application
- high contrast users cannot see many of the toolbar buttons and other user interface elements
- screen reader users cannot interact with the application reliably and effectively, reach and perform the desired functions, and cannot always determine what is being asked in "popup" windows

Because of these and many other problems, which prevent entire populations of people from fully or sometimes even partially using the software, Documents and the Document List cannot be considered accessible.

Introduction

Many higher education institutions are either currently using the Google Apps for Education Suite or are considering adopting the suite for use on their campus. Recently, because of actions of the National Federation of the Blind[1] against New York University and Northwestern University[2], Google has been improving the accessibility of its applications. Google has made significant accessibility improvements to Google Apps, especially in regards to users with visual disabilities. However, it has not made similar improvements for users with other types of disabilities, including mobility impairments and learning or cognitive disabilities. In response, the Access Technology Higher Education Network (ATHEN)[3] formed the Google Apps Accessibility Interest Group (hereafter, Interest Group). ATHEN has a long history of establishing such groups to provide feedback to, and work collaboratively with vendors toward improved accessibility of their products. The Google Accessibility Interest Group consists of accessibility

[1] http://www.nfb.org
[2] http://www.nfb.org/NewsBot.asp?MODE=VIEW&ID=771
[3] http://www.athenpro.org/

experts from a variety of higher education institutions (Appendix A) who performed a set of functional evaluations (Appendix B) to assess whether people with various types of disabilities could perform the functions necessary to effectively use these applications.

Disabilities and Assistive Technologies Tested

Frequently, making a Web page or application accessible to people with visual disabilities will also make it accessible to people with other types of disabilities; however, it does not guarantee that it will. These evaluations considered a variety of disabilities and the corresponding assistive technology used by those populations. They include:

- Visual Disabilities
    - JAWS with Firefox
    - ChromeVox with Chrome
    - Zoom Text with Internet Explorer
    - High Contrast Mode with Internet Explorer
- Mobility Impairments
    - Keyboard-only access with a variety of browsers
    - Dragon Naturally Speaking with Internet Explorer
    - Sticky Keys with a variety of browsers
- Learning or Cognitive Disabilities
    - Read and Write Gold with Internet Explorer

This set of assistive technologies and browsers is not exhaustive, but it represents some of the most common combinations used by people with disabilities. Also, ATHEN did not test all the major screen readers because Google has already stated it only officially supports ChromeVox in Chrome and JAWS in Firefox[4]. While this does not abdicate Google's responsibility for supporting these other screen readers, ATHEN took Google at its word that other screen readers would not work.

Applications Tested

This particular functional evaluation solely examines the Document List and the Documents Application within Google Apps. While the Document List and Documents do seamlessly work together, for purposes of this evaluation it is helpful to consider them as two separate applications. Each one has unique characteristics in how the user interacts with it. Additionally, Google separates the accessibility instructions for them into two different documents.

ATHEN will conduct evaluations on the rest of Google Apps in future work but decided to focus first on Documents and the Document List for two reasons. First, one of the areas of Google's recent accessibility enhancements has been on these applications. This allowed the group to evaluate the point where Google is working the most on improving accessibility and to also see the direction Google is going in its implementation. Second, these applications are some of the most used tools in the Google Apps suite. While GMail and Calendar are also highly used in

[4] http://docs.google.com/support/bin/answer.py?answer=1631663

educational settings, they have not received the recent attention to accessibility that Documents and the Document List have.

Through this evaluation, the group has set up a set of protocols to use for testing other applications in the suite for the next phases of the project.

Technical vs. Functional Evaluations

In assessing a Web application for accessibility, there are two approaches: a technical evaluation or a functional evaluation. A technical evaluation examines the code used in an application and the interactions between the application and the end user to ensure it is implemented according to standards like the Web Content Accessibility Guidelines[5]. It determines if the application is behaving in ways that allow assistive technologies to effectively interact with the system. An example of this type of evaluation is, “Does a particular button in the user interface present itself in a way so that it can be understood by, and interacted using both visual and non-visual means?” The limitation of a technical evaluation is that it does not take a holistic view of asking if the individual pieces of an application work together correctly and if the end user can actually perform the necessary functions with it.

Alternatively, functional evaluations ask if the end user can accomplish the necessary tasks to effectively use an application, regardless of how a particular feature is coded. An example of this type of evaluation is, “Can the user create a new document?” This question is then tested using various modes of interacting with the application, such as using a mouse, a keyboard, or an assistive technology. If the user can accomplish the task through all of those methods, the application is considered functionally accessible for that task.

Functional and technical evaluations are not mutually exclusive. In fact, they help inform each other and both aspects are necessary in developing an accessible application. The individual components of an application have to be implemented to meet a technical standard in order for the functional tasks to even be possible for users of assistive technologies. As the functional tasks are performed and evaluated, a skilled evaluator will be able to readily identify the technical issues behind problems that are discovered.

The analysis in this report concentrates on the functional analysis of Documents for two reasons. First, Documents and the rest of the Google Apps suite are pushing the boundaries of what is possible in a Web browser. Given that reality, Google will need to be creative in how it implements its accessibility solutions and there are multiple technical implementations that can achieve that goal. By concentrating on what can and cannot be accomplished functionally, Google can be made aware of the issues and decide best how to implement that technically. Second, the main goal for the end user is to be able to actually use the application. Instead of getting mired in the details of which technical implementation to use for the solution, ATHEN

[5] http://www.w3.org/TR/WCAG/

can concentrate on the main goal of allowing all users to fully use Google Apps and let Google determine which solution works technically best in its application.

Although there might be multiple possible technical implementations for a solution to an accessibility problem, it does not mean all technical implementations are equally effective. A particular technical implementation might benefit a certain set of disabilities and assistive technologies but might also make it more difficult to find solutions for other types of disabilities. The Interest Group is keenly aware of and concerned about this issue because of Google's concentration on screen reader solutions. The group recognizes the advanced nature of what Google is implementing in its accessibility solution, but by revealing the functional limitations of Google Apps for various types of disabilities, the group wants Google to ask some fundamental questions about its implementation in order to ensure the widest possible number of users can use the Google Apps suite, not just screen reader users.

Results

General Trends

Each disability type and assistive technology tested revealed specific issues, which are detailed in the next section, but the following general trends that adversely impact multiple disability types did emerge:

- Keyboard focus is not always visible
- User interface elements are not visible in all circumstances
- Modal windows allow users to interact with "locked" portions of the application
- Users need to "explore" the user interface outside the standard interaction methods
- Over dependency on shortcut keys
- Inconsistent implementation across browsers
- No ability to apply established Web accessibility standards
- Saving user preferences for assistive technology
- Not utilizing best practices in how assistive technologies interact with applications

Keyboard focus is not always visible

This is important because several assistive technologies rely on input using only a keyboard, or imitating a keyboard input. Navigating items through a keyboard requires being able to visually see where the "focus" is. The focus is a like a pointer that shows the user where their keyboard input will be directed, such as typing text or "clicking" a link.

User interface elements are not visible in all circumstances

"Visible" means different things in different contexts. When viewing the screen in high contrast mode, it means certain items became invisible because the items did not know how to display in that mode. For a screen reader user, it means the screen reader software was not aware of or could not detect the presence of certain user interface elements, and thus could not interact with them.

Modal windows allow users to interact with "locked" portions of the application

Modal windows are the windows that pop up within an application that require a user to interact with it and give some type of input before returning to the main application. A file chooser is an example of a modal window. The way Google Apps implements modal windows, it is quite easy for an assistive technology user to start interacting with the part of the page behind the modal window even while the modal window remains open. Once the user is outside of the modal window, at times it is almost impossible to get back in the window to select "OK" or "Cancel" to close the window. All the while the user thinks they are interacting with the window behind the modal window while not always realizing that their interactions are being ignored by the application. Often the only way to get out of this situation is to reload the page and restart the application.

Users need to "explore" the user interface outside the standard interaction methods

Google has defined a large set of shortcut keys to access all of the functionality of Documents and the Document List. Unfortunately, not all functions are tied to these shortcut keys, or they are implemented incorrectly. This necessitates the user trying other methods of accessing the functions. For keyboard-only users, they use the Tab key to jump from control to control. Screen reader users also must rely on the Tab key and other methods to accomplish some functions. Since these work flows are not the officially supported way of interacting with Documents or the Document List, discovering how to access each part of the user interface is often not intuitive and in some cases can get the user lost and unable to continue working without having to reload the whole page.

Over dependency on shortcut keys

Assigning shortcut keys to every function is one way to give users access to all parts of an application, however, relying on them too heavily can lead to using shortcut keys in instances where they are not needed. Both Web browsers and desktop applications have well established ways of navigating within and interacting with their respective applications and content. Users are familiar with these established methods of interaction and they should be built upon whenever possible in Documents and the Document List.

Inconsistent implementation across browsers

Google has stated that certain browser and screen reader applications are the officially-supported combinations for accessing Documents and the Document List. This is understandable to a degree based on the way Google is implementing accessibility through the Accessible Rich Internet Applications[6] suite (ARIA), which is supported at varying levels in different screen readers and browsers. However, some of the assistive technologies tested should be accessible regardless of whether or not ARIA is used, and thus should not be dependent upon the user's browser preference.

[6] http://www.w3.org/WAI/intro/aria

No ability to apply established Web accessibility standards

The Documents application does not allow one to develop accessible documents. Examples of Web accessibility standards that could not be applied to a document or document element include:

- Alternative text for image
- Table headers or other table accessibility information
- MathML or LaTeX for math equations

Saving user preferences for assistive technology

Google requires JAWS users in Firefox to click a link titled "enable screen reader" in order to enable certain features just for screen reader users. If this link is not clicked, the user will not be able to use Documents. This is problematic because it is not the first link the user hears on the page so the user has to go look for it in each document. It takes pressing the Tab key 18 times after the document loads in order to find and activate this link. Additionally, if it is necessary to only enable certain functionality for screen reader users, this should be a profile-level setting so it is automatically set for users who need it.

Not utilizing best practices in how assistive technologies interact with applications

Based on trends observed so far in how Google is making Documents and the Document List more accessible to screen reader users, the direction of their accessibility implementation is troubling. There are well-established methods for assistive technology users to interact with computer applications, Web pages, and even Web-based applications. Instead of building upon these methods, Google is redefining how screen reader users interact with their computers. Google seems to be relying too heavily on shortcut keys to accomplish almost every task, thus requiring the user to memorize large sets of key combinations. Instead, they should make the user interface itself more explorable and navigable by the user.

Additionally, by concentrating on screen-reader specific solutions and not relying on best practices in accessible Web design, adding support for other assistive technologies might prove to be more difficult. It might require working on a separate solution for each assistive technology rather than one solution for all assistive technologies.

Typically, in accessible Web design, the developer builds an application to a standard, independent of a particular assistive technology's needs. The assistive technology vendors then know how to interact with the application, because they know how to interact with the standard. By defining different ways of interacting with the user interface for different assistive technologies, it will become necessary to support multiple user interfaces for the myriad of possible assistive technologies. It also might require working one-on-one with assistive technology vendors to implement custom solutions. This becomes very burdensome for maintaining applications and often results in certain user groups being left behind as support for their assistive technologies are still being developed while new features are introduced for everyone else.

Specific Assistive Technology Results

The following sections provide high-level overviews of how users were able to interact with Documents and the Document List with different assistive technologies. Appendix C contains more detailed reports for each assistive technology. Additionally, a grade has been assigned to each assistive technology based on the following scale:

- A = a user can fully use all functions of the application
- B = a user can perform most functions using the prescribed methods of interacting with the application
- C = a user can perform many functions, but must rely on non-prescribed methods of interacting with the application
- D = a user can perform some basic functions, but most functions are unavailable or there are other significant problems
- F = a user cannot use even basic functions of the application

An application is neither considered accessible nor providing equitable access until it receives a grade of A.

Dragon Naturally Speaking

Dragon Naturally Speaking[7] (hereafter referred to as “Dragon”) is speech recognition software that allows users to control their computers with voice commands and also to dictate text directly into applications. It is one option for people with mobility impairments.

Grade: F

Documents was completely unusable with Dragon. The most basic of functions such as dictating text into a document and “clicking” links through verbal commands could not be completed.

High Contrast Mode

High Contrast Mode in Windows will increase the font size of text and change the color scheme to provide more contrast between the foreground and the background colors. It is used by some people with visual disabilities.

Grade: D, on Windows; C on OS X

High Contrast Mode had differing results in Windows and OS X. In Windows the text was easier to read on the screen for the most part; however, many of the controls like the menu buttons became completely unreadable in Windows browsers. The button images were no longer visible and hovering with a mouse over the buttons to reveal the tooltips was the only way to discover their function.

[7] http://nuance.com/dragon/index.htm

In OS X the text was also easier to read, but with some notable exceptions. Some of the user interface elements in OS X became difficult to see in High Contrast Mode. The toolbar menus were visible in OS X where they were not in Windows.

Keyboard-Only Access

Keyboard-only access is trying to use an application using only your keyboard without ever relying on a mouse. Keyboard-only access is a critical test because many assistive technologies interface with computers by mimicking the behavior of a keyboard. Also, many users simply cannot use a mouse and need to interact with an application solely through a keyboard.

Grade: C

Many operations could be completed solely through the keyboard such as entering text and basic formatting through the shortcut keys. A major problem, however, is that keyboard focus is often not visible. This is a key component of successfully using an application with only the keyboard. It allows the user to know where their input (typing text, or clicking a button) will be directed. Another major issue in the Firefox browser is that the application menu (File, Edit, etc.) cannot be accessed by only using the keyboard, thus preventing the user from performing many functions in the application.

Sticky Keys

Sticky Keys is not a product, but rather is a function built into operating systems. Sticky Keys allows users to submit key press combinations in serial as opposed to the normal parallel fashion. For instance, instead of simultaneously pressing Control and V, the user can press the Control key, let go of Control, and then the V key. Sticky Keys is used by some people with mobility impairments.

Grade: A, for Sticky Keys itself, independent of keyboard-only issues

Documents and the Document List did not interfere with Sticky Keys; however, all of the problems raised with keyboard-only access also applied to using Documents with Sticky Keys.

ChromeVox with Chrome

ChromeVox[8] is the built-in screen reader for the Chrome browser and the Chrome OS. It is one of the two recommended configurations from Google for screen reader users. Screen readers are used by some people with visual impairments.

Grade: D (could be a C if instability problems are solved)

Most of the basic functions could be performed using ChromeVox through keyboard shortcuts and the application menu, however, ChromeVox was very buggy, requiring frequent restarts of the browser, or reboots in the case of the Chromebook. One notable problem was that it was

[8] http://code.google.com/p/google-axs-chrome/

difficult to navigate to the toolbar menu (e.g., font drop-down menu and formatting buttons) and use it. There was no documentation for how to access these tools and the method for accessing them was discovered by accident.

There were other problems like the inability to read other users' comments and any already existing footnotes. Additionally, there were numerous usability problems, such as the inability to determine when the user has actually finished scrolling all the way to the top of the page or easily getting lost when trying to interact with modal windows.

JAWS with Firefox

JAWS[9] is the most popular Windows-based screen reader. This assistive technology and browser combination is also recommended by Google for people with visual impairments.

Grade: D

While basic text editing was possible in JAWS, many functions were unavailable to the user, and navigating the application was quite difficult. Most of the keyboard shortcut keys worked except for one notable exception - access to the application menu. The standard Alt + Shift + F brought up the browser File menu instead of the Documents File menu. Getting to the Documents File menu required using non-standard methods, such as using the Virtual Cursor. Additionally, there were many usability problems like determining font information about text, reading other users' comments, and knowing when the user had actually scrolled all the way to the top of the document.

Navigation in general with JAWS was quite cumbersome requiring non-standard ways of interacting with applications and requiring users to use JAWS in ways they were not necessarily accustomed to doing. There were also inconsistencies in how the user interface was presented to the user between different tests.

In order for JAWS to function properly in Documents the user must click a link that says "enable screen reader." This is problematic on two levels. First, the link is not the first item the user comes to in the page. It took pressing Tab 18 times to find this link. Second, it should not be required of the user to make this selection every time when this information can be stored in the user profile and automatically executed when needed.

Read&Write Gold

Read&Write Gold[10] is a software application that assists users in reading and composing text through a series of tools such as highlighted word-by-word text-to-speech reading and spell-checking functions designed for particular learning disabilities. It is not considered an alternative means of interacting with an application, like a screen reader. Rather, it adds a layer of functionality on top of an application to aid the user.

---

[9] http://www.freedomscientific.com/products/fs/jaws-product-page.asp
[10] http://www.texthelp.com/

Grade: D

Read&Write Gold does allow users to fully access the functionality of Documents; however, Read&Write Gold's major functionality was not operable in Documents. These functions include reading text, spell checking and dictionary functions. Some functions did work in Documents such as "Speak As I Type" and word prediction.

ZoomText with Internet Explorer

ZoomText[11] is a screen magnification application that allows users to selectively enlarge portions of their screen, such as the area beneath their mouse. It also will read the text from the page to the user.

Grade: A-

ZoomText was able to perform all the functions within Documents when utilizing its screen magnification ability. The only major function that did not perform as expected was its text reading feature, which kept it from scoring a Grade A. Instead of being able to use the standard method of using the arrow keys to start and stop the reading process, the built-in dedicated reading application had to be used.

Conclusion

Google has made significant improvements for screen reader users when using Documents and the Document List, but these tests show that support for screen reader users is still poor and the needs of people with other types of disabilities have not been met. Considerable work still needs to be done to ensure that people with all types of disabilities can fully use the Google Application suite.

Based on the amount of concentrated effort Google has made to Documents and the Document List and the significant shortcomings that still remain, it is likely that all aspects of the Google Application suite will have similar levels of non-support for many types of disabilities.

We hope this report will allow Google to understand the breadth of the problem when designing accessible Web applications and the issues faced by people with varied disabilities and assistive technologies when trying to access their products. While the disabilities and the assistive technologies tested in this report are not exhaustive, they do represent an accurate sampling of the problems faced.

[11] http://www.aisquared.com/zoomtext

ATHEN will be conducting more tests with other assistive technologies and other Google products to better help Google understand what further issues the community of people with disabilities face.

APPENDIX A - Participants

Alice Anderson
University of Wisconsin-Madison

Jon Gunderson
University of Illinois at Urbana-Champaign

Keith Hays
University of Illinois at Urbana-Champaign

Greg Kraus
University IT Accessibility Coordinator
North Carolina State University

Tim Offenstein
Campus Accessibility Liaison
University of Illinois at Urbana-Champaign

Phyllis Petteys
Assistive Technology Specialist
Portland State University

Kevin Price
University of Illinois at Chicago

Hadi Rangin
Information Technology Accessibility & Collaboration Coordinator
University of Illinois at Urbana-Champaign

Karen Sorensen
Instructional Technology Specialist
Portland Community College

Terrill Thompson
Technology Accessibility Specialist
University of Washington

Scott Williams
Web Accessibility Coordinator
University of Michigan

For questions or comments, please contact Greg Kraus (greg_kraus@ncsu.edu), Coordinator of the ATHEN Google Apps Accessibility Interest Group.

APPENDIX B - Functional Task List

1. Document management - completed via the Document List
    a. Create a new document
    b. Open an existing document
    c. Create a new collection
    d. Rename an existing collection
    e. Organize a document into a collection
    f. Rename a document
    g. Share a document with another user
    h. Search the Document List for a file
    i. Download a document as a Microsoft Word file
    j. Delete a document
    k. Obtain a document's properties
        i. Owner
        ii. Last modified date/time, etc.
        iii. Who it is shared with
        iv. Collections it is a part of
2. Document management - completed while editing/viewing the document
    a. Publish a document as a Web page
    b. Share a document with another user
    c. Print a document
3. Basic editing
    a. Enter some text
    b. Read the text
    c. Determine the font style of the text
    d. Set the font style of the text
    e. Determine the font size of the text
    f. Set the font size of the text
    g. Determine the font color of the text
    h. Set the font color of the text
    i. Determine the line spacing of the text
    j. Set line spacing of the text
    k. Create an ordered list with multiple items
    l. Add indention levels to some of the items
    m. Determine what indention level each item is
    n. Determine the justification of the text
    o. Set the justification of the text
    p. Determine the heading level of the text
    q. Set the heading level for the text
    r. Select a portion of the text
        i. Cut the selection
        ii. Paste the selection

s. Copy and paste text from an external text editor or word processing application
t. Navigation within a document
    i. Go to the beginning of a line
    ii. Go to the end of a line
    iii. Go to the beginning of the document
    iv. Go to the end of the document
    v. Go to the next word
    vi. Go to the previous word
    vii. Go to the next sentence
    viii. Go to the previous sentence
    ix. Go to the next paragraph
    x. Go to the previous paragraph

4. Intermediate editing
    a. Hyperlinks
        i. Add a hyperlink
        ii. Modify a hyperlink
        iii. Follow a hyperlink
    b. Images
        i. Add an image
        ii. Modify an image
        iii. Delete an image
    c. Headers and Footers
        i. Insert some text as a header
        ii. Edit the contents of the header
        iii. Insert some text as a footer
        iv. Edit the contents of the footer
    d. Bookmarks
        i. Add a bookmark
        ii. Copy the link for a bookmark
        iii. Delete a bookmark
    e. Comments
        i. Add a comment
        ii. Modify a comment
        iii. Read another user's comment
        iv. Reply to a comment
        v. Delete a comment
    f. Footnotes
        i. Add a footnote
        ii. Modify a footnote
        iii. Read other footnotes
    g. Document Versions
        i. Review an older version of a document though the revision history
        ii. Revert to an older version of a document
    h. Tables

i. Insert a data table
ii. Determine the row and column location for each cell
iii. Modify a table
iv. Delete a table

5. Advanced editing
    a. Multiple Simultaneous Editors
        i. Are you notified of live changes by other users?
        ii. What information, if any, are you able to determine about the changes?
    b. Math Equations
        i. Insert a math equation
            1. Use the following equation: sqrt(a)/b^2 ab2
        ii. Read a math equation
6. Help and Documentation
    a. Find help documentation for Google Docs
    b. Find accessibility documentation for Google Docs

APPENDIX C - Individual Functional Evaluation Test Results

The individual functional evaluation test results are at the following link:

http://athenpro.org/googledocsreports/

The functional task list (Appendix B) was given as a tool for performing the assessment, but not every task will have a corresponding data point, based on the nature of the assistive technology and the way the tester reported their results. Many of the tests were replicated where time and resources allowed.

APPENDIX D - Additional Notes on Documents Accessibility

Documents also provides some additional functionality that has direct impacts on accessibility, like publishing to a Web page, creating PDF files and creating document templates.

Publishing to a Web page

Publishing a document as a Web page introduces numerous accessibility errors:

- The Web page has no doctype.
- The language of the page is not defined, even when the value is set within the document itself.
- Images do not have any alt text or even an alt attribute.
- Tables do not have necessary headers, captions or summaries.
- Unordered lists <ul> are stored as ordered lists <ol>, substituting bullets for numbers.
- Lists more than one-level deep output deeper nested parts of the list as entirely new lists instead of nested lists. CSS and the start attribute are used to give the illusion that it is one continuous list.
- CSS is used instead of semantic markup for common HTML tags such as:
    - bold
    - italics
    - superscript
    - subscript
- Equations are presented as images with no alternative text.

Creating PDF files

PDF files created within Documents are untagged documents. Tags are critical to making PDF files accessible. Without tags, Adobe Acrobat has to make best guess attempts at tags, which are not always accurate.

Creating document templates

Document templates are a starting point for new documents that contain pre-populated content. Templates cannot be edited by users, but rather are copied into a new document for further editing. Document templates neither add nor remove any accessibility features beyond what a standard Document already has.

APPENDIX E - Alternative Ways to Interact with Documents

There are options for interacting with Documents other than using Google's Web interface. The primary methods use Microsoft Word to edit the document and then using some type of synchronization method to put the changes back into Documents. Because Microsoft Word has been made quite accessible over the years, offers an opportunity for an accessible alternative to the Web interface. Three methods were examined as alternatives: uploading and downloading Microsoft Word documents, OffiSync and Google Cloud Connect.

Uploading and Downloading Microsoft Word Documents

This technique requires the user to manually download the Document as a Microsoft Word Document, make changes within Microsoft Word and then re-upload the file to Google Docs.

This method works quite well with the document retaining all of its data except for the following problems:

- Alternative text entered in Word is removed when uploaded to Documents.
- Equations created in Documents appear as meaningless text in Word.
- Equations created in Word appear as an image with no alternative text.
- Any table header information entered into Word is removed in Documents.
- End notes in Word are converted to footnotes in Documents.

The biggest problem with this technique is it does not allow for multiple simultaneous editors. This is because a document edited in Word and re-uploaded is uploaded as a new document, not a revision to a previous document.

OffiSync

Offisync is an add-on for Word that appears as a new tab within the ribbon. It allows for live editing in Word of a document stored online within Google Docs. It also allows for multiple live editors where changes are synchronized across all the editors. In practice, the synchronization is unusable for live multiple edits. Edits would take anywhere from a few seconds to two minutes to propagate. If more changes were made in the interim period, then some items might be overwritten in the synchronization. When the files synchronize in Microsoft Word, the changes can be brought in with the Track Changes feature so the user can review the changes. If the changes are rejected, they are also removed from the online version of the document stored in Google Docs.

Similar to uploading and downloading a Microsoft Word document, this technique works quite well in terms of retaining the document structure and formatting. The biggest problems are as follows:

- Alternative text entered in Word is removed when uploaded to Documents.
- Equations created in Documents appear as meaningless text in Word.
- Equations created in Word appear as an image with no alternative text.
- Any table header information entered into Word is removed in Documents.
- End notes in Word are converted to footnotes in Documents.
- Comments made with Word do not pass through to the online Google Document.

Also of note, some items, like images, could duplicate themselves within the document when multiple users are editing from within Microsoft Word.

Google Cloud Connect

Cloud Connect is a Google product that lets you automatically synchronize Microsoft Word documents among multiple users via Google Documents. When the file is stored in Google Documents it is stored as an actual Microsoft Word file, so it is read-only through the Web interface. This will greatly limit how other users can interact with the document. The synchronization of changes across multiple users is quite responsive. Some of the major problems with this technique are:

- Alternative text entered in Word is removed when syncing to Google Documents.
- Equations created in Word appear as an image when synced to another user.
- Tables could have some formatting issues at times when syncing with another user.

Conclusion for Alternative Ways to Interact with Documents

The appropriateness of these alternative methods will vary depending on the need of the user. If multiple live editors are needed, then none of the alternatives will meet the need. If there will only be one editor of a document, OffiSync works fairly well since it keeps most of the document in tact and it allows automatic synchronization with Google Docs. In the end, none of the alternative methods provide even close to the same experience and functionality as accessing a document through Google's Web interface.