Building graphical development tools for the visually impaired

Tanya Lung
Department of Computer Science,
University of Saskatchewan
Saskatoon, Canada, S7N 1L7
1 306 966-4886
tzt178@mail.usask.ca

ABSTRACT

Software development is a complicated process. Development tools, while allowing basic functions needed for programming and design, often provide advanced features such as diagramming and 'drag and drop' form creation. While these tools make development much more efficient for the average developer, the visual nature of these tools is aimed at a sighted audience. This creates barriers for the visually impaired. To overcome these barriers guidance is introduced for improving usability for the visually impaired. The guidance addresses diagramming and looks at how DBVisAssist, a previously created graphical development tool, is successful in overcoming these barriers.

1. INTRODUCTION

Computer Science is a popular major for disabled students [3][4]. This makes sense because computers are improving many of their lives. It seems reasonable that the industry which brought them new abilities should also be the industry where it is easiest to work, however, this is not the case.

Franqueiro and Siegfried state that "[w]hile there have been many efforts to make the World Wide Web and visual representations of data more accessible to the blind, there have been fewer efforts to make it easier for the blind to program beyond the accessibility hardware and software that facilitate their use of computers in general"[13]. This has become such a problem that one software engineer interviewed by Horstmann et al said that she "had been made redundant when her department switched to UML, as she was unable to visualize the diagrams"[14]. These problems need to be alleviated. DBVisAssist, a previously created graphical development tool, tries to overcome some of these problems. This paper will introduce guidelines and how they affect DBVisAssist.

2. BACKGROUND

When designing and implementing complex systems, programmers often employ development applications, also known as Integrated Development Environments (IDE), which simplify the processes of designing, programming, and debugging. Development speed is hastened because "…the trend has been for developer tools to provide more automatic creation of major elements of applications"[7]. Some IDEs provide features for diagramming, documentation and collaboration with colleagues. These programs increase sighted programmers efficiency but, since they are visual, can hinder efficiency of a visually impaired programmer. Franqueiro and Siegfried point out that when computers were mainly command line based, visually impaired users where on more equal footing with their sighted counterparts. The movement to Graphical User Interfaces (GUIs) has put the visually impaired at a disadvantage when using these interfaces as well as when creating them[13].

When designing products to be used by people with various abilities the question often becomes how much should one support? Perhaps Vanderheiden said it best when he stated:

> "No single interface technique will work. Creating an everyone interface sounds wonderful, but it can sound unobtainable. Trying to design to a single least common denominator interface clearly does not work. If we use only those abilities or input techniques that everyone has and can use in any environment we would have to rule out all visual, auditory, and tactile interfaces."[29]

Vanderheiden suggests that one should increase the number of supported users by supporting assistive technologies. He claims that the "[k]ey to achieving everyone interfaces is the provision of all basic information in either a modality-independent or a modality-parallel (flex-modal) form"[29]. Modality-independent refers to information stored in a format not specific to any one mode of presentation (visual, auditory or tactile). Vanderheiden recommends ASCII text because it can be easily presented in all three modalities. Modality-parallel refers to providing multiple modalities which can work together or separate.

Since visually impaired programmers require extra support when learning how to program[2][3][4][6][12], it is this author's standpoint that an application be built with the visually impaired as it's main audience.

2.1 Development Applications

There are many different IDEs available. Some of the more common development applications are Visual Studio and Eclipse. Both of these applications have highly intense visual components involving multiple frames and toolbars.

Currently the visually impaired are using Assistive Technologies (AT) to use the same developer tools as sighted programmers. Interfacing development applications with AT isn't always the easiest, cost effective and most efficient solution.

Screen readers are an AT tool used by the blind for navigating computers. Unfortunately screen readers can fail when accessing computer programs designed without screen readers in mind. Franqueiro and Seigfried point out that problems arise when working with development applications "…in which most programmers "point and click" to design the forms on which their applications rely"[13].

For example, on the American Foundation for the Blind website, one programmer talks about his encounters with certain developer tools. He complains that the "… Visual

FoxPro development environment was nearly impossible to use with a screen reader"[1]. He goes on to talk about how "[Visual Fox Pro] lacked keyboard access to some features, used nonstandard controls, and some text was invisible to some screen readers"[1]. This programmer complained about Visual Basic 6.0, indicating icons in the toolbox are invisible to most screen readers. The only screen reader able to handle the toolbox was Window Bridge and it could only read the default set of icons. Otherwise, visually impaired users must use "help" for each icon before they can determine the icon they are on.

Califf et al found in their first year classes use Eclipse that although essential functions were usable for a visually impaired student, "Eclipse is not ideal; there are several functions that [the student] cannot access"[3].

2.2 Assistive Technologies

Cohen defines AT as "… a general term used to describe devices or software that helps an individual provide input or receive output from the system"[4]. There are currently many ATs available for the visually impaired. It should be mentioned that ATs are often expensive and can cost thousands of dollars.

2.2.1 Screen readers

Cohen et al state that "[t]he main technology that visually impaired users use to access a computer is a *screen reader"*[4]. Screen reader software runs behind other applications and reads text that appears on the screen. Screen readers are unable to provide context and cannot read images (unless underlying coding provides a description i.e. an alt tag). Examples of some common screen readers are JAWS, Window-bridge, and Window-eyes. Basic screen readers are now being implemented into Operating Systems such as Windows Vista and Mac OS.

2.2.2 Screen magnifiers

Screen magnifying software is used by visually impaired who have some sight capabilities but require text and images to be enlarged. Examples of common screen magnifier programs are Zoomtext, LunarPlus, and MAGic.

2.2.3 Braille printers and Embossers

Braille printers or embossers use a number of techniques to print Braille to paper. Braille can be embossed onto paper via impact (pins creating indents in the paper) or by using swell paper (paper which swells when heat is applied).

2.2.4 Refreshable Braille display

A Braille display can connect to a computer in order for the user to read in Braille what appears on their computer screen. The device works by raising and lowering pins to provide the Braille output. As a user moves around the screen, the display automatically updates itself.

2.2.5 Optical Character Recognition & Scanners

Optical Character Recognition is used by visually impaired to scan hard copies of documents to turn into digital documents. This enables the user to output the document into a more usable format such as Braille or voice.

2.2.6 Closed-Circuit Television (CCT)

A video camera projects a magnified image onto a screen. Video monitors, televisions, and computer monitors can be used as the screen.

3. WORKING WITH DIAGRAMS

This paper focuses on the use of diagramming in IDEs and the barriers it creates for visually impaired programmers. .

3.1 Diagramming Tools

Some development applications allow the ability to create diagrams pertaining to the project in development. A simple example of this is the ER-Diagramming feature available in MS Access.

For the sighted, diagrams assist with understanding relationships between objects. These diagrams provide a method for planning projects and provide a reference for the development and maintenance of these projects. Diagrams are highly visual and as such, diagramming tools are built with sighted users in mind. Horstmann et al mention that "[a]ccess to diagrams is currently provided to blind people in the form of either verbal descriptions or tactile diagrams"[14].

When examining the research, diagramming tools seem to be split into two different types, programs that allow users to read diagrams and those that allow users to draw.

3.1.1 Reading

In their research, Horstmann et al identify three different types of diagramming techniques for the blind. These techniques are:

1. Tactile diagrams combined with touchpad technology "… are sometimes referred to as audio-tactile diagrams"[14]. This involves placing a tactile diagram onto a touchpad. When the diagram is touched, auditory feedback is provided.
2. Combining refreshable displays with sound.
3. Translating visual images to auditory images "for example using the pitch and timbre of different musical instruments to indicate different aspects of the image"[14].

AudioGraph [18] investigated the use of a touch panel in conjunction with auditory feedback. Visually impaired users interact with the system by touching the panel to select parts of a diagram. Each selection is displayed to the user aurally. For example, a connection sounds like a plucked string, and text is verbalized. Although users were able to read the diagrams, users sifted through an excess of information because everything is explained in extensive detail.

iGraph-Lite[10][11] focuses on making graphics accessible in Statistics Canada's publication "The Daily". The project focuses on generating summaries of graphical data and exploring data by the use of sound. The project provides interaction with the use of natural language. Users can navigate and explore data otherwise represented in graphical format only. Graphics are input into the system as XML files. In this case most XML files are an export from a MS Excel document. The XML is analyzed and input into the navigator which generates natural language to describe the graphs. However, the reliance on natural language and the simplistic command structure make it difficult for the user to customize.

TeDUB[14] describes diagrams and focuses on Unified Modeling Language (UML) diagrams. The diagram descriptions are based on guidelines presented by the Confederation of Transcribed Information Services (COTIS). Bitmap images or text-based electronic data format are acceptable input methods. Users navigate via keyboard or

joystick. This solution allows collaboration between sighted and non-sighted users. Another strength is the flexibility of the system; users create their own methods for examining diagrams. There are some downfalls such that "[t]he image analysis stage is prone to errors that can lead to inaccurate interpretation and failure to identify important image components"[14]. Their solution involves sighted users supervising and intervening when needed which removes independence from the visually impaired user.

PLUMB [2][5][6][7] is one of the few tools which supports both reading and creating graphs. Their focus is on "communication of graphs and relational information to blind computer science students"[7]. Graphs are displayed on the tablet PC and, with the help of auditory cues, blind users navigate graphs using either the tablet and pen and/or a keyboard. Calder et al mention "[t]he downside is that exploration can be slow and depends on the precision of the users hand movements"[7]. As a result, it is "…difficult to get information about incident graph elements since the user needs to move the pen around the area and wait for sound notification"[7].

PLUMB uses XML documents with the Graphics eXchange Language (GXL) to create graphs. Users create graphics via the command line or with second party GXL supported programs. Unfortunately, the paper does not provide information regarding feedback during diagram creation.

3.1.2 Creating

Kamel and Landay explain that "[b]lind users have had only limited success in using drawing programs because traditional drawing software lacks the capability to translate graphical data output in a way that screen access programs can interpret"[16]. They go on to say that in her paper, "Millar showed the importance of visual feedback in a study comparing the ability of congenitally blind and sighted but blindfolded children to draw the human figure"[16]. Kamel and Landay also state, as suggested by Millar, that detail provided in these drawings was directly related to visual feedback. It was also mentioned that "[s]ighted but blindfolded children performed only slightly better than congenitally blind children in creating details, cohesion, and alignment"[16].

Some drawing solution involve touchpads combined with tactile images. Kamel and Landay explain "this requires the user to purchase a tablet, possibly a prohibitive expense"[16]. Instead, they use electronic images making images easier to modify. The Integrated Communication to Draw (IC2D)[16] tool uses grids to provide visual element locations. A keyboard provides input and output is provided aurally. Users draw images in a 3x3 recursive grid. A recursive grid refers to the ability to subdivide a cell into another 3x3 grid up to an additional two levels; providing up to a 27x27 grid. Kamel and Landay found that levels beyond 27x27 become too difficult to conceptualize. Locations on the grid are presented in the form of a telephone keypad because blind individuals are trained on the use of a telephone keypad in school. Users specify their own labels for objects and positioning. Although it gives users a more precise, feedback oriented drawing system, it is time consuming and not diagram specific.

Kurze created TDraw [19], which places heat-sensitive, swell paper onto a Thermostifft digitizing tablet. Input is recognized as users draw on the swell paper and provide appropriate voice commands. Users connect attributes to the elements they are drawing by providing verbal commands. Once a drawing has been completed, a tablet and special pen can be used to explore. The computer recognizes when the pen approaches an element and provides text-to-speech output to the user. Unfortunately, the swell paper means that drawings cannot be altered later. As well, there is no feedback provided to the user during the drawing process. This makes it difficult to draw items in relation to one another for example, two shapes of equal size.

3.2 Mental Models

Humans are visual by nature. We have art such as paintings, movies and literature. All these items traditionally require sight. A fair assumption is that someone who has never had sight probably thinks and deals with things differently that the average sighted individual.

3.2.1 Perceiving Spatial Relationships

Visuospatial perception refers to the ability to recognize and visualize spatial relationships and locations between objects. Cornoldi found that "numerous studies showed that congenitally totally blind people are able to generate and process visuospatial images and in some cases, the blind can perform as well as sighted individuals" [8]. In explanation of his findings, Cornoldi presents the concepts of passive storage and active storage. Passive storage is the retention of visuospatial information while active storage is the transformation, manipulation, or integration of stored memory. He found that visually impaired individuals perform passive tasks involving visuospatial images with the same accuracy as sighted people, but had more trouble than sighted people when required to manipulate the images. Cornoldi directly relates this problem to his finding which indicate "congenitally blind cannot have visual traces" and "cannot use traces to generate more complex mental images" [8]. This is because visual traces, a type of mental imagery, are gained only by visual experience.

3.2.2 Visual Tracing

Cornoldi [8] states that visual traces are different from generated mental images which are gained from other information such as haptics and long term memory. Orienting in space can be easy, however updating mental representations can be challenging. Visually impaired individuals can make use of mental imagery strategies similar to sighted individuals. Visually impaired people tend to work best within two dimensional spaces where they can perform better than the sighted in memory tasks.

3.2.3 Learning

Schneider [24] suggests that it is best to use the learning-by-doing principle. He goes on to explain that when visually impaired people are about to embark on an outing to a destination they have never visited before, they often memorize the layout of a given area, learn the pathway segments, and then the angles between the paths [24].

Sanchez and Aguyo observed that blind learners rely on past experience to build their abstract thinking. Only time using their program helped users gain the necessary experience for understand the programming paradigms.

Jeung and Gluck [15] conducted experiments using muti-modal feedback when dealing with thematic maps. In a number of different experiments, information was provided to

participants for two variables. The information was provided in one of the following combination of modes: both variables as visual information, one variable as visual and one as auditory, one as visual and one as haptic, and finally one as auditory and one as haptic. Interestingly, they found that when asked to remember information, people using multimodal display did best when the display combined auditory and haptic feedback.

In Horstmann et al's study they determined that visually impaired users could build a mental representation of hierarchical structures as well as navigate them. It was also noted "…how visually impaired users prefer to conceptualize hierarchies based upon their existing experience of navigating tree structures in programs such asWindows Explorer"[14]. This means that they preferred to navigate information from left to right.

3.2.4 Sensing

Lai and Chen conducted an interview where 'listening to music' was one of the most popular methods for passing time. Interview and experimental results led them to the conclusion "…that hearing is one of the most essential channel toward external messages"[20]. As a result they suggest that further studies should be done on the interfaces between humans and audio appliances.

Two-point tactile threshold felt by ones hands was another area of experimentation conducted by Lai and Chen. They found that the blind's fingertips have a sharper sensation than a normal-sighted person's fingertips. They concluded "this probably has a lot to do with the fact that the blind students are accustomed to using the pulp of their index finger to read in Braille"[20].

3.3 Input and Output

As mentioned earlier, visually impaired persons learn by building on their previous capabilities and experiences. Combining the aforementioned mental model with Vanderhieden's suggestion about modality-independent and modality-parallel, we can see that there are three basic input methods; keyboard, voice, and touchpad with tactile overlay. A system can work functionally with the use of keyboard or voice input for the majority of visually impaired users. A touchpad on it's own however, doesn't provide the same capabilities. Touchpads will be discussed later as they are both a source of input and a source of output.

The different methods of output available to the visually impaired are audio and tactile. Audio can be provided as speech output or a number of variously mapped sounds. Since auditory output is the most common method for receiving output, this is one medium which must be supported. Tactilely, users can access Braille displays, Braille printers, and touchpads with tactile overlays.

A touchpad requires the user to have some knowledge and context about a system before using it. It is easy to hypothesis that the reasoning behind PLUMBs slow diagram exploration times is due to the following reasons:

1. Users with no diagram experience would have difficulty when first trying to understand a diagram.
2. Touchpads do not provide tactile feedback making it easy for a user to become disoriented.
3. It difficult to locate and relocate specific parts of drawings on a touchpad. Without sight, one can only use some method of estimation to determine exact locations.

Using a tactile overlay helps elevate most, if not all, of the above mentioned problems. However, since not all visually impaired users are familiar with Braille and tactile diagrams, the design of these tactile overlays will need to be generic. As users will probably require training in order to recognize certain elements.

It is the recommendation of this author to provide all information at least aurally. Since multi-modal interaction achieves the best memory results, it would be beneficial to provide tactile feedback as a secondary channel.

3.4 Barriers

When embarking on a journey by foot, if you need to cross a river without a bridge, the river becomes a barrier. Similarly, computer users can encounter barriers. Donker et al[9], identified four barriers to accommodate for when designing software for visually impaired users.

1. The pixel barrier refers to screen output being in pixel maps which screen readers are incapable of reading.
2. The graphics barrier refers to the loss of information during translating images into words. It often results in long, complicated textual descriptions.
3. The layout barrier refers to using layout for semantic reasons.
4. The mouse barrier refers to the visually impaired being unable to use a mouse as an input medium.

This paper proposes a fifth barrier, the drag and drop barrier. This refers to tasks that, while easy to complete using drag and drop or even point and click, are highly complex without the use of that modality.

3.5 Creating and Reading Diagrams

As mentioned earlier, Horstmann et al designed their system around guidance provided by COTIS. These guidelines are important when determining how to display your diagrams verbally to users.

1. State why the diagram is there.
2. Differentiate between different levels in diagrams.
3. Describe visual aspects of the diagram. Use terms such as 'egg shaped', vertical, and perpendicular.
4. Summarize the diagram.
5. Make important details apparent.
6. Minimize interpretation.
7. Order descriptions; provide start points and direction.
8. Include all features of the diagram.
9. Identify components, clarify labeling and identify differences.

It is important for the user to understand whether a relationship exists between 'table 1' and 'table 2'. When creating diagrams to share with sighted individuals, it may also be important for the visually impaired person to have an understanding of the layout. Then they can change the diagram as suggested by peers and understand discussions if someone says something

like “the table that appears in the top right hand corner”. It may also be important to allow the user to choose his start points in the diagram.

To output the diagram information aurally, consider the following strategies in addition to the guidelines presented by COTIS. These strategies should help overcome barriers and achieve the highest understanding of the user:

1. Stay away from similar sounds [9].
2. Use contextual reinforcement for relationships [9].
3. Use non-pixel mapped graphics [9].
4. Keep descriptions short and precise [9].
5. Divide information that requires processing into subparts [8].
6. Use chunking to reduce the amount of to-be-treated elements [8].
7. Use strategies for working with visual manipulation to overcome task difficulties [8].
8. Provide an overview, then offer further details [26].
9. Allow users to locate and relocate important points and “[s]ome guideline or frame of reference is needed”[16].
10. Make use of 3x3 grids where possible [16][17][26]. to increased accuracy for visually impaired when locating objects. The use of recursive grids can achieve more detail however, a 27x27 grid should be the maximum size used.
11. Allow the user to primarily navigate diagrams using the keyboard. Secondary navigation can be provided via a tactile diagram overlaid on a touchpad.
12. Present line diagrams as a "layered" sequence of diagrams [12].

Creating diagrams should be kept simple. To accomplish this, I introduce three guidelines in combination with two guidelines from other sources.

1. The system should automate as much of the process as possible.
2. Due to input modality constraints, all the input should be handled via keyboard however, voice may be a modality to be explored further.
3. Users should be offered a 3x3 grid in order to control the location of objects [16][17][26].
4. Offer users preference options before rendering the drawing. These preferences should include items such as location of objects, the level of the drawing (detail offered), objects to appear in the drawing.
5. Allow the user to easily toggle the tasks of reading and creating/modifying the diagrams [17].

4. DBVisAssist

DBVisAssist [22] is currently a prototype, previously built by this author which allows the visually impaired to create Entity-Relationship diagrams.

DBVisAssist presents and obtains information using a series of webpages. These pages are presented to the user in a task-based succession but the user can deviate from the path at any time. A system of simple links is used to navigate through DBVisAssist.

The system utilizes four strategies for visualization. These include using chunking, creating subparts, employing visual manipulation strategies, and providing overviews. Four strategies are used for obtaining information aurally: providing contextual reinforcement, providing short descriptions, not using image files and staying away from similar sounds.

4.1 Chunking and Subpart Dividing

DBVisAssist decomposes into two parts: one part allows adding data for ER-Diagrams and the second part provides visualization and navigation of the ER-Diagrams.

Users traverse through a series of tasks to add data to create an ER-Diagram. The system presents the tasks in sequence, which the user can easily follow. An example of a task would be adding a table and it’s attributes.

To visualize and navigate through an ER-Diagram, users are provided with a verbal description of the ER-Diagram. Division of information and diagrams themselves into subparts allows users to retrieve information from the lowest to highest levels of detail. The subparts, listed from the lowest level to the highest, are:

1. Read a single table and its attributes
2. Read a single table and its relations
3. Read all the tables, relations, and locations
4. Read the entire diagram

4.2 Spatial Manipulation.

Allowing users to change table locations within a diagram requires spatial manipulation. In order to provide the user a method to deal with manipulation, a grid system similar to IC2D is used. IC2D assumes most users are familiar with a telephone keypad; however, DBVisAssist assumes that software developers would expect to use a keyboard number pad layout.

Each region in the grid is given a number based on the keyboard keypad. Each table is assigned to a region meaning the system can currently support up to 9 tables. The system automatically assigns each table to an optimal position and automatically switches table locations when the user moves a table into an occupied region.

Locations of tables in the diagram are important for understanding how the outcome may appear sighted users or when collaborating with sighted users. The DBVisAssist system reports the location of a table in the diagram grid. For example, “Table Teachers is located in region 1”. Descriptions of the diagram are read in order of the regions (1-9). They begin with the location of the table, move on to the table’s attributes, and then the relations. The descriptions do not describe the diagram exactly because it is unnecessary to explain the actual visual appearance. Instead, only necessary information is conveyed (the tables, attributes, locations, and relationships).

4.3 Contextual Reinforcement

As suggested by Filepp, users should not be required to remember sequences of information. Therefore, relationships are presented using prose. For example, if the user inputs a

relation named “Teach” between the teacher and student tables, the relation is read back as “Table Teachers teach Table Students”.

However, if a user does not input a relation name, the system automatically assigns the relation name of “is related to” so the relation is read as “Table Teachers is related to Table students”. When reading tables and attributes, the system explains each item such as “Table Teachers” or “Attribute firstname”.

4.4 Overviews

Each task provides an overview that describes what has already been done and/or what the system will be doing. For example, if the user is adding a new table, the system lets them know the number of existing tables and the name of each table.

4.5 No Image Files

In order to avoid image files, diagrams need to be created using a markup language. It is necessary that the screen-reader reads a description of the diagram and not the diagram itself. Since the diagrams are readable by the FireVox screen reader, there can be elements that do not need to be read to the reader.

5. EVALUATION

My current research evaluated DBVisAssist in three areas: web accessibility, correct visual representation of an ER-Diagram, and a navigable verbal representation of an ER-Diagram.

The World Wide Web Consortium's Web Content Accesiblity Guidelines (WCAG) [28] identify three increasing levels of accessibility, referred to as Priority 1, 2, and 3. WebXACT [27] is a tool for testing web accessibility baased on WCAG’s guidelines.

I created an ER-Diagram using DBVisualizer, an IDE used with various DBMSs, and compared it to the visual ER-Diagram created by DBVisAssist.

Initial user testing was performed on DBVisAssist’s navigational scheme with two sighted, one visually impaired, and two simulated visually impaired participants (who used DBVisAssist without a screen). Simulated visually impaired participants were used because of the difficulty in finding visually impaired people trained computer science [3][25]. It should also be mentioned that the results obtained from the simulated visually impaired subjects should be similar to someone who became visually impaired later in life.

Participants were given brief training on using the screen reader and DBVisAssist. Using the think-aloud method, users performed a number of tasks. Some tasks required the input of information. Tasks were split into two categories; creating diagrams and reading diagrams. Tasks for creating diagrams were: determine the number of tables in the system, determine the table names, add tables, add relationships, and move tables. The tasks for reading the diagrams were to determine the attributes in a table, determine the relationships of tables, and to determine the location of tables. Post-testing interviews were conducted to determine users perceptions of the system.

To determine if the tool was navigable by visually impaired and sighted users, I used two variables; efficiency and error. An efficiency measure of 0-2 was used, where 0 meant the user failed to complete the task, 1 meant the user completed the task but took a longer method to get to the final result, and 2 meant the user took a path which lead directly to the correct answer. The error rate was based on the amount of wrong paths taken. Errors specifically related to the screen reader (such as using wrong commands) were ignored because the screen reader's usability wasn’t being tested.

6. RESULTS

Each page of DBVisAssist was tested in WebXACT. The application successfully achieves a Priority 3, meaning that it achieves the highest level of accessibility as recommended by WCAG guidelines.

When comparing the diagrams from DBVisAssist against the diagrams from DBVisualizer, results reveal minor differences. The largest difference is that DBVisAssist displays names of the relations where Dbvisuallizer displays datatypes. These differences can be seen in Figures 1 and 2.

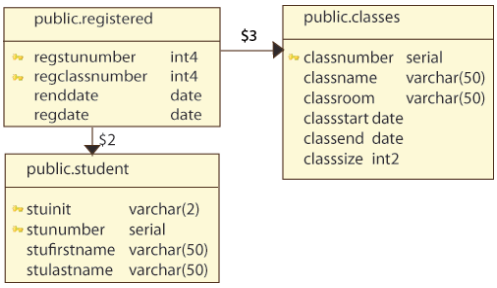

Figure 1. ER-Diagram from DBVisualizer

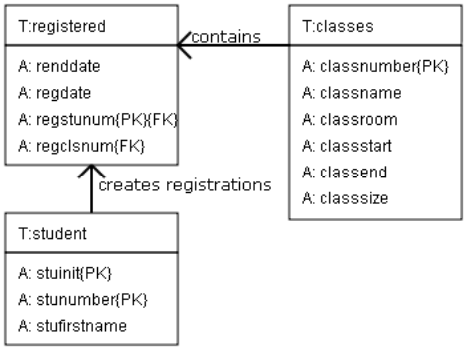

Figure 2. ER-Diagram from DBVisAssist

The sighted and simulated visually impaired participants indicated the diagrams from DBVisAssist were basic but comparable to other programs they used. One sighted participant had experience with four other diagramming tools. The participant indicated that DBVisAssist was the easiest diagramming tool she had used. These comparisons show the diagrams from DBVisAssist are acceptable to sighted readers.

The simulated visually impaired participants listened to the auditory diagram. When shown the corresponding visual diagram, they commented that the visual diagram was similar to what they expected.

All participants successfully navigated through DBVisAssist with only minor difficulties attributed to learning the screen reader. In fact, tasks involving finding and reading the required diagrams had no errors for both the visually impaired and sighted groups. However, the efficiency for visually impaired and simulated visually impaired participants was not as good as for the sighted. There were three instances where participants took an unexpected path but were still able to find the correct information.

Interestingly the visually impaired subject had 0 errors which was better than even the sighted subjects who had a sum of 3 errors. The simulated visually impaired subjects had a sum of 10 errors. The tasks with the most errors were the "adding tables" task with 7 errors and "add relationships" task with 4 errors. Table A illustrates the errors by user for tasks: A: determine the amount of tables in the system; B: determine all the table names; C: add tables; D: add relations; E: move tables; F: determine the attributes in a table; G: determine the relations of a table; and H: determine the location of a table.

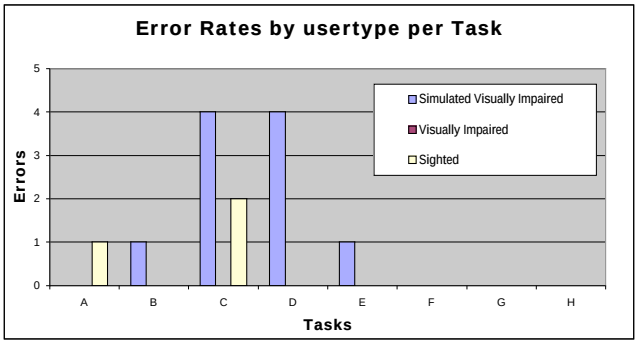

Table 1: Error rates by user type per Task.

The lack of errors during reading the diagrams shows that DBVisAssist helps with understanding relationships between objects within a database regardless of the user's sight abilities and that the diagrams are easy to navigate.

For efficiency, both sighted participants were 100% efficient, the visually impaired participant and one simulated visually impaired participant were 89% efficient, and the remaining simulated visually impaired participant was 72% efficient.

An analysis of errors and of post-testing interviews reveals that tasks should be broken down further. For example, the location information could be removed from the diagrams and included in its own section. Participants indicated that they were more interested in the relationships then the locations of the tables when they were reading these diagrams. The location information seemed to be important only if they needed to modify the diagram.

Some users suggested they liked the amount of detail provided by the system while performing tasks while others indicated that they needed either more or less detail. One participant wished the tool adapted to the user's experience with the program by offering less detail once the user was more experienced. This suggests that incorporating user preferences would be beneficial.

The visually impaired user seemed excited and mentioned that he hadn't heard of a similar tool. He was hopeful that further work would be done so that the system could support tactile graphics and other screen readers.

7. Conclusion

DBVisAssist achieves goals of providing a tool that allows sighted and visually impaired users the ability to share diagrams.

The diagrams are visually and aurally readable. The oral descriptions allow the user to visualize the relationships belonging to a database. The accessible tool allows the users to create diagrams and control the locations tables appear in the diagram. The diagrams, though basic were comparable to diagrams created by another database diagramming program. These results along with our participants opinions showed that the diagrams were acceptable.

Finally, DBVisAssist proved navigable by sighted and visually impaired users. Users were able to navigate the program and specially the diagrams with few errors. When comparing the error rates between sighted and visually-impaired users, there were minor differences.

8. Future Work

DBVisAssist is a good first step towards creating an accessible IDE. However, it currently does not take into account all guidance suggested in this paper. Interestingly, the unaccounted for guidance issues were also brought up by the participants. These items include: compatibility with existing assistive technologies, layering the diagrams more, allowing bookmarks and allowing preferences to be set for your diagram.

Currently, DBVisAssist is capable of displaying up to a maximum of nine tables. Since many complex systems contain more tables, it would be beneficial for DBVisAssist to support more tables.

9. REFERENCES

[1] American Foundation for the blind. 2008. The Right (or Required) Tool for the Job: Microsoft Developer Tools and Screen Readers. accessed on April 2, 2008 http://www.afb.org/Section.asp?Documentid=1411

[2] Calder, M., Cohen, R. F., Lanzoni, J., Landry, N., and Skaff, J. 2007. Teaching data structures to students who are blind. In Proceedings of the 12th Annual SIGCSE Conference on innovation and Technology in Computer Science Education (Dundee, Scotland, June 25 - 27, 2007). ITiCSE '07. ACM, New York, NY, 87-90.

[3] Califf, M. E., Goodwin, M. M., and Brownell, J. 2008. Helping him see: guiding a visually impaired student through the computer science curriculum. In Proceedings of the 39th SIGCSE Technical Symposium on Computer Science Education (Portland, OR, USA, March 12 - 15, 2008). SIGCSE '08. ACM, New York, NY, 444-448.

[4] Cohen, R. F., Fairley, A. V., Gerry, D., and Lima, G. R. 2005. Accessibility in introductory computer science. In Proceedings of the 36th SIGCSE Technical Symposium on Computer Science Education (St. Louis, Missouri, USA, February 23 - 27, 2005). SIGCSE '05. ACM, New York, NY, 17-21.

[5] Cohen, R. F., Haven, V., Lanzoni, J. A., Meacham, A., Skaff, J., and Wissell, M. 2006. Using an audio interface to assist users Who are visually impaired with steering tasks. In Proceedings of the 8th international ACM SIGACCESS Conference on Computers and Accessibility (Portland, Oregon, USA, October 23 - 25, 2006). Assets '06. ACM, New York, NY, 119-124.

[6] Cohen, R. F., Meacham, A., and Skaff, J. 2006. Teaching graphs to visually impaired students using an active auditory interface. In Proceedings of the 37th SIGCSE Technical Symposium on Computer Science Education (Houston, Texas, USA, March 03 - 05, 2006). SIGCSE '06. ACM, New York, NY, 279-282.

[7] Cohen, R. F., Yu, R., Meacham, A., and Skaff, J. 2005. PLUMB: displaying graphs to the blind using an active auditory interface. In Proceedings of the 7th international ACM SIGACCESS Conference on Computers and Accessibility (Baltimore, MD, USA, October 09 - 12, 2005). Assets '05. ACM, New York, NY, 182-183.

[8] Cornoldi, C. (2000). Mental imagery in blind people: the role of passive and active visuospatial processes. In Heller, M. A. (Ed), Touch, Representation, and Blindness. (pp.143-181) Oxford University Press, NY: New York.

[9] Donker, H., Klante, P., and Gorny, P. 2002. The design of auditory user interfaces for blind users. In Proceedings of the Second Nordic Conference on Human-Computer interaction (Aarhus, Denmark, October 19 - 23, 2002). NordiCHI '02, vol. 31. ACM, New York, NY, 149-156.

[10] Ferres, L., Verkhogliad, P., and Boucher, L. 2007. (Natural language) interaction with graphical representations of statistical data. In Proceedings of the 2007 international Cross-Disciplinary Conference on Web Accessibility (W4a) (Banff, Canada, May 07 - 08, 2007). W4A '07, vol. 225. ACM, New York, NY, 132-133.

[11] Ferres, L., Verkhogliad, P., Lindgaard, G., Boucher, L., Chretien, A., and Lachance, M. 2007. Improving accessibility to statistical graphs: the iGraph-Lite system. In Proceedings of the 9th international ACM SIGACCESS Conference on Computers and Accessibility (Tempe, Arizona, USA, October 15 - 17, 2007). Assets '07. ACM, New York, NY, 67-74.

[12] Francioni, J. M. and Smith, A. C. 2002. Computer science accessibility for students with visual disabilities. In Proceedings of the 33rd SIGCSE Technical Symposium on Computer Science Education (Cincinnati, Kentucky, February 27 - March 03, 2002). SIGCSE '02. ACM, New York, NY, 91-95.

[13] Franqueiro, K. G. and Siegfried, R. M. 2006. Designing a scripting language to help the blind program visually. In Proceedings of the 8th international ACM SIGACCESS Conference on Computers and Accessibility (Portland, Oregon, USA, October 23 - 25, 2006). Assets '06. ACM, New York, NY, 241-242.

[14] Horstmann, M., Lorenz, M., Watkowski, A., Ioannidis, G., Herzog, O., King, A., Evans, D. G.,Hagen, C., Schlieder, C., Burn, A. -M., King, N., Petrie, H., Dijkstra, S. and Crombie, D. (2004) 'Automated interpretation and accessible presentation of technical diagrams for blind people', New Review of Hypermedia and Multimedia, 10:2, 141 - 163

[15] Jeong, W., and Gluck, M., (2002) "Multimodal bivariate thematic maps with auditory and haptic display", Proceedings of 2002 International Conference on Auditory Display, Kyoto, Japan, pp. 1-4.

[16] Kamel, H. M. and Landay, J. A. 2000. A study of blind drawing practice: creating graphical information without the visual channel. In Proceedings of the Fourth international ACM Conference on Assistive Technologies (Arlington, Virginia, United States, November 13 - 15, 2000). Assets '00. ACM, New York, NY, 34-41.

[17] Kamel, H. M. and Landay, J. A. Sketching images eyes-free: a grid-based dynamic drawing tool for the blind. In *Proceedings of the Fifth international ACM Conference on Assistive Technologies* (Edinburgh, Scotland, July 08 - 10, 2002). Assets '02. ACM Press, New York, NY, 2002, 33-40.

[18] Kennel, A. R. 1996. Audiograf: a diagram-reader for the blind. In Proceedings of the Second Annual ACM Conference on Assistive Technologies (Vancouver, British Columbia, Canada, April 11 - 12, 1996). Assets '96. ACM, New York, NY, 51-56.

[19] Kurze, M. 1996. TDraw: a computer-based tactile drawing tool for blind people. In Proceedings of the Second Annual ACM Conference on Assistive Technologies (Vancouver, British Columbia, Canada, April 11 - 12, 1996). Assets '96. ACM, New York, NY, 131-138.

[20] Lai, Hsin-Hsi. "A study on the blind's sensory ability." International journal of industrial ergonomics 36.6 (2006):565-570.

[21] Lin, C., Francioni, J. M., Hossain, A., and Kang, P. 2004. Accessible student-directed visualization of computer organization concepts. In Proceedings of the 2004 OOPSLA Workshop on Eclipse Technology Exchange (Vancouver, British Columbia, Canada, October 24 - 24, 2004). eclipse '04. ACM, New York, NY, 47-51.

[22] Lung, Tanya. 2006. DBVisAssist: Creating ER-Diagrams for the Visually Impaired. Cmpt 898. Accessible Computing.

[23] Sánchez, J. and Aguayo, F. 2005. Blind learners programming through audio. In CHI '05 Extended Abstracts on Human Factors in Computing Systems (Portland, OR, USA, April 02 - 07, 2005). CHI '05. ACM, New York, NY, 1769-1772.

[24] Schneider, J. and Strothotte, T. Constructive exploration of spatial information by blind users. In Proceedings of the Fourth international ACM Conference on Assistive Technologies (Arlington, Virginia, United States, November 13 - 15, 2000). Assets '00. ACM Press, New York, NY, 2000, 188-192.

[25] Smith, A. C., Francioni, J. M., and Matzek, S. D. 2000. A Java programming tool for students with visual disabilities. In Proceedings of the Fourth international ACM Conference on Assistive Technologies (Arlington, Virginia, United States, November 13 - 15, 2000). Assets '00. ACM, New York, NY, 142-148.

[26] Strain, P., McAllister, G., Murphy, E., Kuber, R., and Yu, W. 2007. A grid-based extension to an assistive multimodal interface. In CHI '07 Extended Abstracts on Human Factors in Computing Systems (San Jose, CA, USA, April 28 - May 03, 2007). CHI '07. ACM, New York, NY, 2675-2680.

[27] Watchfire Corporation (2003), WebXACT, Retrieved March 20, 2007, from http://webxact.watchfire.com/

[28] W3C (2006), Web Content Accessibility Guidelines (WCAG). Retrieved, December 22, 2006, from http://www.w3.org/WAI/intro/wcag.php

[29] Vanderheiden, G.C., and Henry, S.L., Everyone Interfaces User Interfaces for All: Concepts, Methods, and Tools. Edited by Constantine Stephanidis. Lawrence Erlbaum associates, Publishers. Mahwah, New Jersey, 2001. Pg 115–13