A New Approach towards Visual Programming for the Blinds

Mario Konecki*
* Faculty of Organization and Informatics,
University of Zagreb, Pavlinska 2, 42000 Varaždin, Croatia
E-mail(s): mario.konecki@foi.hr

Abstract - Visual user interfaces have left blind programmers that were able to perform programming job for years in a challenging situation. This problem has especially been present in classical desktop programming due to the nature of technology used in this field. Efforts have been made to help blinds work with graphical interfaces but none of them gave a fully usable solution for blind programming professionals. To solve this issue several different approaches can be applied but only one of them can lead to generally usable solution. In this paper mentioned approaches are discussed and proposed solution is described in more detail in the form of GUIDL (Graphical User Interface Description Language) that represents a new system that would include a new description language and would enable blinds to make graphical interfaces as a part of their programs in a way that would be acceptable to not only professional programmers but also to designers and other blind computer hobbyists. The demands for this kind of system as a combination of initial ideas and results obtained by research among the blind programmers will also be presented. Finally, a formal description of proposed system and its parts including a description language as its core part will be given.

I. INTRODUCTION

Blinds have been the significant part of computer technology revolution since its very beginning. In 1970s computers have found its appliance in many business systems and other areas of life. Personal computer started to be something that is an essential part of people's life. This huge variety of possible computer users have made a huge difference in the way that blinds study, work and communicate with their colleagues. The main aiding technologies that made this possible were Braille typewriter and text-to-speech or speech-to-text synthesizers that made reading and using various computer programs accessible to the blinds.

These technological means have greatly supported a greater percentage of successful blinds that were able to study or work not only in the field of computer science but also in other areas. Text has become available and new information was no longer an issue for the blinds. The very same aiding technology has virtually enabled blinds to do the work of professional programmers. This was not a problem because all software was text based and developed speech synthesizers could support all necessary actions. Inclusion of blinds into the world of computer science and professional programming was done and the interest of the blinds for this area of work never faded [1].

The same aiding technology enabled blinds to become professional programmers and since all software was text based this wasn't a problem. Speech synthesizers were able to cope with all software and programming was brought to the blinds as an equal opportunity. The inclusion of blinds into programming was successful and this interest never faded [1].

Although inclusion of blinds in the world of programming was successful it wasn't long lasting. 1980s and development of Graphical User Interfaces (GUI) brought a number of problems for blind computer users and programming professionals. The prospect of existing technologies for aiding blinds was not bright and it was obvious that existing methods won't be sufficient in the very near future.

Several issues have emerged. Software development has become so rapid that it stopped being documented in the way that would be satisfied for the blind professionals. Existing aiding technologies weren't enough to support a huge variety of new concepts and different software. GUI adoption made the worst problem since speech synthesizers could not even cope with all graphical elements and were not able to visualize the graphical layout of the screen. These tools tried to read text from certain graphical controls with more or less success but the general context was in many cases lost. One of the things that called for a suitable solution was the problem of designing graphical elements that became a part of almost every piece of software. Point and click visual environments weren't suitable for the blinds and textual description of these environments became virtually impossible [2]. All this has left blind professionals in a very hard position.

Importance of blinds inclusion into the world of professional programming can be confirmed today by over 130 registered blind programming professionals listed at American Foundation of the blind programmers and also by some promising data that can be seen in other countries. Programming is listed as one of popular occupations for blinds in Czechs Republic, Finland, Poland and Spain [3] and it's also stated as one of promising new carrier opportunities for the blinds in Europe [4].

II. CURRENT EFFORTS AND ISSUES

Aiding tools today are mostly based on the same technology that was used from the very beginning and that is speech-to-text and text-to-speech converters. The existing converters and similar in nature and differ in details and some specialized functions. The most known and used speech synthesizers today are:

- JAWS [5]
- HAL Screen Reader [6]
- COBRA [7]
- Window Eyes [8]
- Easy Web Browsing [9]

All these tools have similar problems among which are problems with graphical representation of images and large amounts of tabular data and also the problem of robustness or inability to support new technologies and concepts in sufficient amount. There are also efforts aimed at learning of programming concepts rather than supporting professional activities. One of these efforts can be seen in development of Audio Programming Language (APL) tool [10] and JavaSpeak [11]. There are also some interesting conceptual attempts to visualize graphical shapes to the blinds like Audiograph [12] which uses different sounds and pitches of sounds to represent different shapes and coordinates. An approach that combines speech, music, tactile panels, Braille and other media to represent visual concepts and images to the blinds is called multimodal approach [13].

Another interesting area where the problem of visualization is present is World Wide Web. Web pages have become very graphically and multimedia oriented which also made their translating into text very difficult and sometimes impossible. The fact that makes this issue even worse is the fact that over 50% of web pages tend to be partially accessible on inaccessible to the blinds [14]. This shows that web developers don't know much about web pages accessibility or think that it is of no great importance. Tactile panels and tactical web browsing [15] are technological attempts to elevate this issue. Tactical web panels are multiline Braille panels that try to represent image by using a series of pins that can be lowered or elevated (Figure 1 [15]).

All these efforts offer some features and solve some issues but none of them offer a usable solution for blind programming professionals. If we look at the two main areas of computer programming it can be concluded that these issues are not so prominent in the area of web programming.

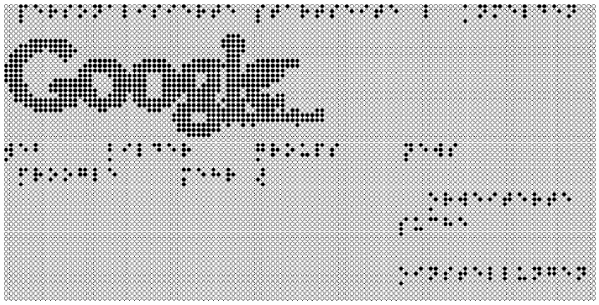

Figure 1. Tactical display

This is because of the nature of web technologies and the fact that blinds are still able to code web programs through pure text which is completely supported by existing speech synthesizer's software. There are some minor issues with color recognition and some other accessibility elements [16] but these are issues on the lower levels. Main issues arise in the area of GUI programming where almost all environments adopted graphical interfaces and complex environments that are not adequately supported by existing aiding tools. So, it can be concluded that the area of designing GUI is the area that needs to be focused on and addressed in order to provide blinds with means to be once again fully included in the world of professional programming.

III. TOWARDS SOLUTION OF BLINDS ISSUES

In order to address and solve mentioned issues of blind programmers several approaches can be undertaken.

- Interpreters that would support different types of controls and interpret different form elements and attributes could be developed.
- Programming environments could have built-in support for designing of graphical elements through speech.
- Specific scripting languages could be developed that would enable textual creation of specific elements in specific programming languages. An example of this kind of scripting language can be found for Visual Basic forms [17].

Finally in order to create a general solution a more abstract approach can be taken. A new description language can be developed would enable creation of different User Interface (UI) elements in a simple and understandable way. This kind of specification would then be translatable to a particular programming language GUI code using specific GUI generator. In this way the generally usable and extensible solution would be proposed that would be usable to programmers but also to pure blind designers. This kind of system with an new description language as its central part would be called Graphical User Interface Description Language (GUIDL). The benefits of this kind of solution would be in inclusion of blinds in the overall process of designing and exchanging ideas among the whole team and in enabling blinds to design their own final solutions as a equal part of development team.

IV. REQUIREMENTS FOR GUIDL SYSTEM

The first step in the development of the new GUIDL system is to define the requirements which such system would have to satisfy. Based on the literature analysis and research several requirements can be identified that this kind of system would have to include in order to be usable and sustainable.

Mentioned requirements are:

- Easiness of usage: system has to be easy to use for programmers but also to other interested professionals such as designers
- Intuitive, simple and easy to understand syntax: descriptive language that will be used for description of user interfaces graphical elements has to be intuitive and easily applicable to all that are in the business of programming classical GUI applications but also to other programmers and designers.
- Independence of descriptive language: descriptive language of UI definition has to be general and not programming language specific or programming environment specific.
- Extensibility of descriptive language: descriptive language has to be extensible in the way that it enables inclusion of new programming environments and languages and in ideal case inclusion of new graphical elements or controls.

In order to evaluate stated requirements and to find out about the interest of blind users for proposed solution a suitable research has been conducted. The research has been conducted through specifically designed questionnaire that included statements needed to assess the mentioned requirements but also questions regarding other aspects and elements that have been recognized as potentially relevant in the process of designing the questionnaire. Open type questions were also included so that blinds would have chance to make their own comments and state additional aspects that possibly were not included as initial requirements.

In order to conduct the research in the satisfying way it was necessary to find appropriate places where it would be possible to address the desired population. Several well known groups of blind programmers have been identified and included into research scope [18] [19] [20] [21] along with some less known lists. This population represented a convenience sample that was addressed with the e-mail request to participate in the research.

In the moment of research analysis the 37 answers were obtained and it can be expected that this number will be even higher in the future regarding that the research is still ongoing. Overall results showed that blind population is generally interested in this kind of solution. A part of research that was of great value was the part that included personal comments.

Respondents were given a possibility to apply with their e-mail for future testing of the system prototype which has been announced to be available in the near future. Respondents have also been instructed that additional research will be conducted to assess the usability of developed system prototype according to defined requirements and project goals. Most interesting question and their average values are given in Table 1. All answers were graded using Likert scale from 1 to 5 (1 - I disagree completely, 5 – I agree completely).

Table 1. Research results

Statement	Average reply value	Standard deviation
How old are you?	28,51	6,22
How long have you been programming?	7,64	4,64
Designing Graphical User Interfaces is one of the biggest issues for me as a programmer.	4,59	0,79
The issue of Graphical User Interfaces design is more prominent in the area of desktop programming than in the area of web programming.	3,91	1,11
I would like to have a solution to the issue of designing Graphical User Interfaces.	4,70	0,66
It is important to me to be able to create interfaces that can be included in standard programming environments (Visual Basic, C#, etc.).	4,35	0,78
It is important to me to be able to describe visual interfaces in simple and familiar way.	4,54	0,69
I would like a new programming environment and new programming environment rather than including my designs in existing programming environments.	1,43	0,76
I don't care how complicated is to describe visual interfaces, if it works.	1,78	0,85
I would like the language and system used to describe visual interfaces to be very easy to use with simple syntax.	4,51	0,60
I would like the language and system used to describe visual interfaces to have intuitive and recognizable syntax similar to some well known programming language.	4,16	0,72
I would like the language and system used to describe visual interfaces to be independent of particular programming environment and translatable to several well known programming languages environment's forms.	4,48	0,69
I would like the language and system used to describe visual interfaces to be extensible and have an ability to include new programming languages when needed.	4,02	0,86

Respondents were also asked about their experience in programming and whether they have more experience in web or classical desktop programming or both. There were 29 male and 8 female respondents and 11 respondents were primarily desktop oriented, 8 web oriented and 18 respondents stated that they have experience in both desktop and web programming.

From the comments of respondents along with other results can be concluded that they consider important the simplicity of the syntax and user interface along with proper documentation as one of very important aspects. All these results have been taken into consideration as requests for GUIDL system.

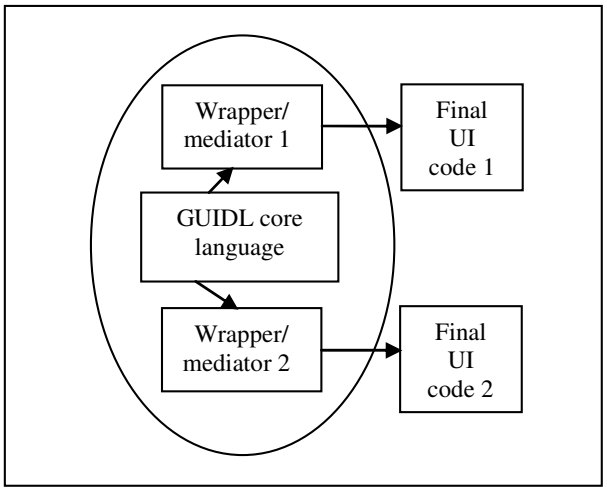

Figure 3. GUIDL conceptual model

V. GUIDL MODEL OF DEVELOPMENT

After defining the requests for the GUIDL system it is necessary to define the model and parameters of GUIDL development process. The conceptual model of GUIDL is shown in Figure 3.

GUIDL system consists of several elements. The core part is GUIDL central description language. GUIDL language serves for description of graphical interface and GUIDL source code is stored for future processing. GUIDL code is processed after UI description by one of specific mediators. Every mediator serves as a translator of GUIDL UI code to a programming language specific UI code. In this way the extensibility of the system is supported. The first step in the process of GUIDL language development is to define the parameters in the sense of scope and syntax of the language and to define other aspect of the language. In the language prototype some restrictions have been made in the sense of supported graphical controls. Selected graphical controls represent the most used controls in the applications development.

Selected controls are:

- Form
- Button
- Label
- TextBox
- CheckBox
- ComboBox
- PictureBox
- RadioButton

In the same way certain control properties were selected. This initial graphical control set will be expanded regarding future research results. Syntax of GUIDL language is designed in the way to be simple and of known form. The language that best fulfills these criteria is Beginner's All-purpose Symbolic Instruction Code (BASIC). GUIDL language is because of that inspired by BASIC but does not have any direct link to it and BASIC interpreter/compiler cannot understand GUIDL code. Syntax inspired by BASIC assures simplicity and easiness of usage that is important in order for GUIDL to be used not only by programmers but also by designer and other computer user.

Example of GUIDL code along with partial grammar in EBNF is given below.

```
Form 'form name'
  Size = 'width' 'height'
  Location = 'x coordinate' 'y coordinate'
  Caption = 'form title'
  WindowState = 'visual state of a form on
  runtime'
  ControlDeclaration
End
```

EBNF grammar:

```
form =  'Form ', formname, formattributes,
controldeclaration, 'End';
formname = alphabeticcharacter, {alphabeticcharacter
| digit}, eol;
formattributes =  sizeattribute, locationattribute,
captionattribute, windowstateattribute, eol;
sizeattribute = 'Size = ', width, ws, height, eol;
```

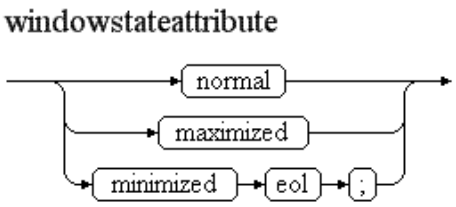

Figure 4. Syntax diagram of windowstate atribute

```
locationattribute = 'Location = ', xcoordinate, ws,
ycoordinate, eol;
caption = alphabeticcharacter, {alphabeticcharacter |
digit}, eol;
windowstateattribute = 'normal' | 'maximized' |
'minimized', eol;
controldeclaration = {controldeclaration}, eol;
alphabeticcharacter = "A" | "B" | ... | "W" | "X" | "Y" |
"Z" ;
digit = "0" | "1" | "2" | "3" | "4" | "5" | "6" | "7" | "8" |
"9" ;
ws = ? white space character ? ;
xcoordinate = digit, {digit}, eol;
ycoordinate = digit, {digit}, eol;
eol = ? CR (carriage return) character ? ;
```

Syntactical diagram of windowstate attribute is given in Figure 4.

After the grammar of GUIDL is defined the development of GUIDL language and system will be conducted according to defined structure of GUIDL which is given in Figure 5.

The GUIDL source code is processed by GUIDL lexer that conducts the lexical analysis of the code.

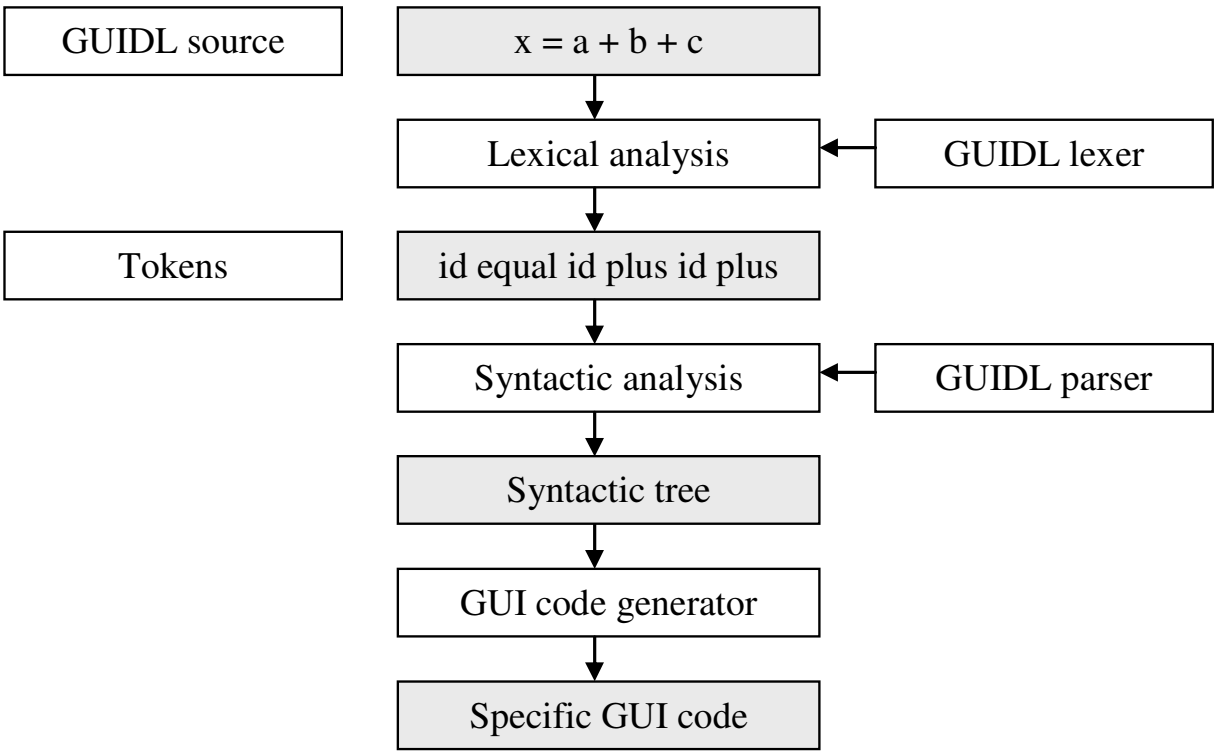

Figure 5. Structure of GUIDL

Constituent part of lexer is the scanner. Scanner reads the file with the GUIDL source code character by character and forwards it to lexer. Lexer accepts the signs and indentifies acceptable character strings as tokens. The set of possible tokens is defined inside of GUIDL parser form where it is included into lexer. GUIDL parser reads tokens recognized by lexer and conducts the syntactical analysis based on grammar rules of GUIDL. GUIDL parser creates syntactical trees that will serve as basis for specific GUI generator that will translate GUIDL code into programming language specific code.

After the development of initial prototype further testing and modification of GUIDL model will be conducted according to research results.

VI. CONCLUSION

Blinds have been using computers since the very beginning of computer era. They have performed programming jobs with great success using various aiding technologies available. Since GUI development took place blind programmers have found themselves in difficult position particularly regarding development of GUI. Existing aiding tools weren't able to support different GUI development environments in a way that would be generally usable to blind programmers and existing efforts didn't give conclusive results.

In order to solve this problem several approaches can be undertaken which include development of more sophisticated aiding tools, integration of different aiding technology to development environments and development of programming language specific scripting languages. In order to provide a generally usable solution a new model of GUIDL description language and system has been proposed based on defined requirements obtained by research that would provide simple and extensible solution for development of GUI. Testing and improving of GUIDL system will be a part of future research efforts.

REFERENCES

[1] A. Steve, "Blind Programmers Face An Uncertain Future", ComputerWorld, p. 1, 1998.

[2] K. G. Franqueiro and R. M. Siegfried, "Designing a Scripting Language to Help the Blind Program Visually", ACM Proceedings of the 8th international ACM SIGACCESS conference on Computers and accessibility, ACM Press, pp. 241-242, 2006.

[3] R. Cattani, "The employment of blind and partially-sighted persons in Italy: A challenging issue in a changing economy and society", available at http://www.euroblind.org/media/employment/employment_Italy.doc, Accessed: 25th March 2011.

[4] bfi Steiermark, "European Labour Market Report", available at http://eurochance.brailcom.org/download/labour-market-report.pdf, Accessed: 25th March 2011.

[5] B. J Rosmaita, "Accessibility First!: a new approach to web design", ACM Proceedings of the 37th SIGCSE technical symposium on Computer science education, ACM Press, pp. 270-274, 2006.

[6] I. J. Pitt and A. D. N. Edwards, "Improving the usability of speech-based interfaces for blind users." In Proceedings of the Second Annual ACM Conference on Assistive Technologies (ASSETS), 1996.

[7] "Screen Reader COBRA", available at http://www.baum.de/cms/en/cobra/, Accessed: 25th March 2011.

[8] "GW Micro - Window-Eyes", available at http://www.gwmicro.com/Window-Eyes/, Accessed: 25th March 2011.

[9] "Easy Web Browsing", available at http://www-03.ibm.com/able/accessibility_services/EasyWebBrowsing.html, Accessed: 25th March 2011.

[10] J. Sanchez and F. Aguayo, "Blind learners programming through audio", ACM CHI '05 extended abstracts on Human factors in computing systems, ACM Press, pp. 1769-1772, 2005.

[11] A. C. Smith, J. M. Francioni and S. D. Matzek, “A Java programming tool for students with visual disabilities”, Proceeding Assets '00 Proceedings of the fourth international ACM conference on Assistive technologies, ACM Press, 2000.

[12] J. Alty and D. Rigas, “Communicating Graphical Information to Blind Users Using Music: The Role of Context”, CHI '98 Proceedings of the SIGCHI conference on Human factors in computing systems, ACM Press/Addison-Wesley Publishing Co., pp. 574–81, 1998.

[13] Y. Bellik and D. Burger, “Multimodal interfaces: New solutions to the problem of computer accessibility for the blind”, CHI '94 Conference companion on Human factors in computing systems, ACM Press, pp. 24–28, 1994.

[14] T. Sullivan and R. Matson, “Barriers to use: Usability and content accessibility on the web’s most popular sites”, Proceedings of the ACM Conference on Universal Usability, ACM Press, pp. 139-144, 2000.

[15] M. Rotard, C. Taras, and T. Ertl, “Tactile web browsing for blind people”, Multimedia Tools Appl., Springer, 37(1):53–69, 2008.

[16] K. L. Spencer, “Assessing the Accessibility for the Blind and Visually Impaired of Texas State Agency Web Sites”, Texas State University-San Marcos, Political Science Department, Public Administration, Texas State University, 2001.

[17] R. M. Siegfried, “Visual programming and the blind: the challenge and the opportunity”, SIGCSE '06 Proceedings of the 37th SIGCSE technical symposium on Computer science education, ACM Press, pp. 275-278, 2006.

[18] http://www.freelists.org/list/program-l, Accessed: 5th February 2012.

[19] [31] http://www.freelists.org/list/programmingblind, Accessed: 5th February 2012.

[20] http://BlindGeeks.org, Accessed: 5th February 2012.

[21] [33] http://www.freelists.org/list/jawsscripts, Accessed: 5th February 2012.