Volume 4, Issue 11, November 2014 ISSN: 2277 128X
International Journal of Advanced Research in Computer Science and Software Engineering
Research Paper
Available online at: www.ijarcsse.com

Design and Development of Integrated Development Environment with Data Prefetching and Complexity Parameter

Parul*, Jagandeep Sidhu
Computer Science and Engineering, Chitkara University,
Punjab, India

Abstract*— The integrated development environment (IDE) with data prefetching and complexity parameter can help the programming beginners as well as professionals to fastly grasp the basic language grammars and the drawing functions of languages like C, C++. The objective function and constraints of the problem model through data prefetching is to enhance learning efficiency of the beginners as well as helpful tool for the professionals. Input process output cycle of an IDE includes a text editor with certain navigation buttons and menus, a GNU compiler collection (GCC) is to be linked to the IDE and an output is to be presented using graphical user interface (GUI) components. This interactive IDE provides a platform for programming developers to develop various applications and also help to reduce the cost of programmer cultivating. One of the key ingredients in our approach is the representation of time and space complexity.

Keywords*— Integrated Development Environment; C Language; C++ Language; Data Prefetching; Complexity Parameter

I. INTRODUCTION

This article is concerned with the construction and development of a tool known as the IDE. Other terms, such as Interactive Development Environment and Integrated Design Environment, are also used to refer to IDE's. An IDE is computer software that provides comprehensive facilities to the programmers for the development of the software. It generally consists of a source code editor, a compiler or interpreter or both, build automation tools and a debugger. Ideally, use of an IDE results in greater productivity as it presents a single program in which all development is done [1][2][3]. Languages like C, C++ are the basis course in colleges and universities now. Many students are using Turbo C or VC++ for developing applications which leads to low learning efficiency of the beginners as the visualization is poor, the operating is not convenient, errors are not found easily and development functions are expanded difficulty. Though, we can develop applications on Code Blocks, Visual Studio, and Borland C++ Builder and so on but still tools are not supporting graphics functions [4].

II. DESIGN AND DEVELOPMENT OF IDE

The interface of an IDE is shown as Fig. 1.

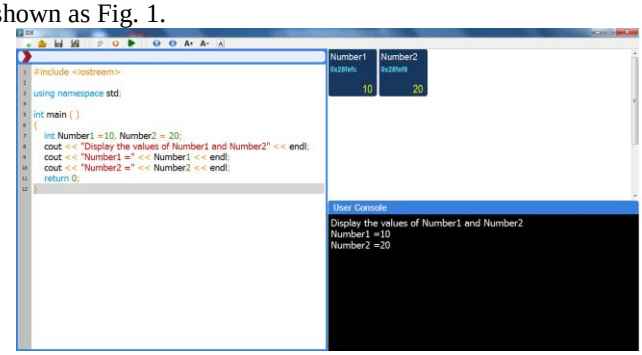

Fig. 1 IDE Function Interface

A. Design and Development of the User Interface

The Graphical User Interface (GUI) is layered out as Fig. 2 after abstracting [4].

<table>
<tr><td colspan="2">Main Window</td></tr>
<tr><td colspan="2">Toolbar</td></tr>
<tr><td rowspan="2">Source Code Editor</td><td>Graphics View</td></tr>
<tr><td>User Console</td></tr>
</table>

Fig. 2 Graphical User Interface layout

GUI of the IDE is formed by using two layouts Vertical and Horizontal and by some widgets such as Frame, Plain Text Edit, Label, Graphics View, Horizontal Scroll Bar, Vertical Scroll Bar. Horizontal and vertical layouts ensure that widgets within are aligned horizontally and vertically respectively [5]. Frames are the placeholder in which widgets like Plain text Edit, Label, Graphics View, Horizontal Scroll Bar, Vertical Scroll Bar etc can be placed [6]. Plain Text Edit is an advanced editor used to handle large documents and responds quickly to user input [7]. Label is used to display text or image [8]. Graphics View is used for managing large number of 2D graphical items [9]. Horizontal Scroll Bar and Vertical Scroll Bar allows user to access parts of document that is greater than the widgets [10].

B. Functions of the IDE

The toolbar is formed by using icons like 1 New which is used to open a new file, 2 Open to open a already saved file, 3 Save to save a file, 4 Save as to save a already saved file, 5 Build to compile a file, 6 Build and Run to compile and run a file, 7 Refresh to move to first line of code, 8 Line Up to move up line by line, 9 Line Down to move down line by line, 10 Font Increase to increase the font of the code, 11 Font Decrease to decrease the font of the code.

Basically, Main Window is divided into three parts Source Code Editor, Graphics View and User Console. Source Code Editor is a plain text editor where user can write a code. Graphics View is used to display the address allocated to each variable as well as value of each variable using graphical items like rectangle. User Console not only display the output of entire program but also display the output at each line of code.

C. System Construction Based on the Plug-ins

The IDE features can be extended by using plug-ins. The objective of plug-ins is to expand and strengthen the software functions in the case of not modifying the program (the platform). The IDE is designed using a plug-ins framework so as to make it easy for users to improve it without affecting the code written on source code editor [11]. The IDE contains five plug-ins Syntax Highlighting, Parenthesis Matching, Line Numbering, Line Highlighting, Line Description. The plug-ins are shown as Fig. 3.

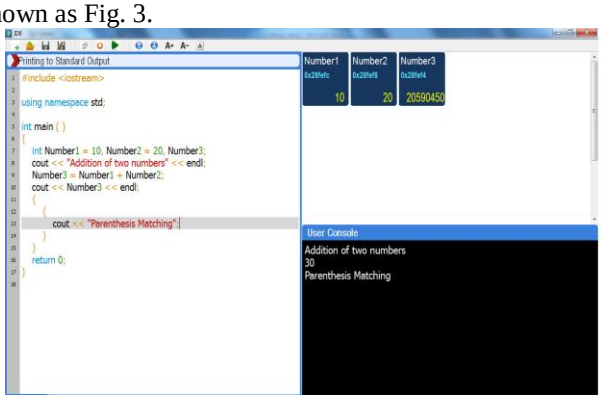

Fig. 3 IDE Plug-ins

Syntax Highlighting is used to highlight parts of the text written on a plain text editor by using certain colours like blue is for keywords, orange is for operators. It helps the user to read the text and identify syntax errors [12]. Parenthesis Matching or Braces Matching is used for matching sets of braces (square brackets, curly brackets or parenthesis and small brackets). Each line of the code is assigned with consecutive line numbers and Line Numbering is used to highlight the current line of the code with line number. As Fig. 3 shows that the source code editor displays the line numbers in an area to the left of the area for editing. In Line Highlighting, the source code editor will highlight the line containing the cursor [13]. Line description provides line by line description of the code.

D. Linking with GCC

The compiler is of course the core tool [14]. Translation Hierarchy to convert the source code into executable code is shown as Fig. 4.

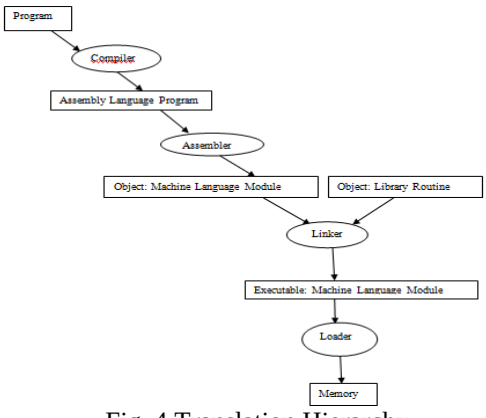

Fig. 4 Translation Hierarchy

First of all, the compiler translates the source code into object code. The object code is not yet executable. The next step is to convert the object code into executable code that is accomplished with the help of linker. Linker will combine the object code with library routine to form executable code which will be loaded into the memory with the help of loader. GCC is one of the packages of Minimalistic GNU for windows (MINGW) which consists of a complete Open Source Programming tool set that helps in the development of windows applications. Other packages included in MINGW are binutils like assembler and linker, headers and libraries, windows API header files and libraries, GNU debugger and GNU make [17].

GCC commands are used to link a compiler with the IDE. The command to compile and link source file input.c into executable file output.exe is gcc –o output.exe input.c. The command to compile and link source file input.cpp into executable file output.exe is g++ –o output.exe input.cpp [14][15][16]. Standard Input and Output Redirection is a powerful mechanism for redirecting input, output and errors. Operators of Standard Input and Output Redirection are shown as TABLE I.

TABLE II. STANDARD INPUT AND OUTPUT REDIRECTION OPERATORS

Operators	Action
>	Redirect standard output
<	Redirect standard input
2>	Redirect standard error
>>	Append standard output
<<	Append standard input
2>>	Redirect and Append standard error
>&	Redirect both standard output and standard error

Standard Input and Output Redirection is of three types 1 Standard Input (stdin) which is the source of input data for text mode programs. 2 Standard Output (stdout) in which streams of data produced by text mode programs. 3 Standard Error (stderr) which is the destination of error messages from text mode programs [18][19]. Fig. 5 shows streams of error messages from text mode programs after compilation.

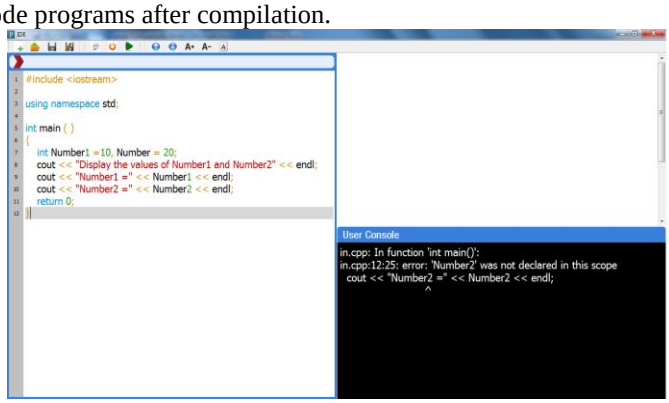

Fig. 5 Error Messages produced from program after compilation

E. Functions of Data Prefetching

The IDE using data prefetching technique to display the address allocated to each variable as well as value of each variable using graphical item on graphics view and also help to display the output at each line of code on user console. Data prefetching is an effective way to evaluate data access latency caused by cache misses and to bridge the performance gap between memory and processor. Data prefetching technique is used for hiding data access latency, which is used to unlink and overlaps data transfers and computation. Data prefetching technique first of all predicts future data accesses, then starts a data fetch, and lastly brings the data to the processor before it is requested [20]. Address and value allocated to each variable is shown as Fig. 6.

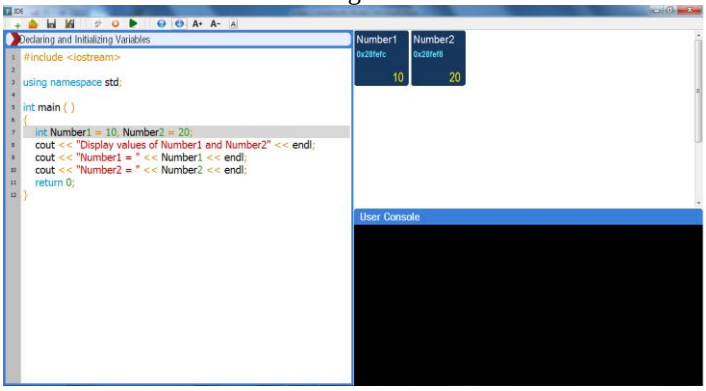

Fig. 6 Address and value allocated to each variable

Fig. 7 and Fig. 8 are used to show the line by line output. Fig. 7 shows the output till 9th line of code.

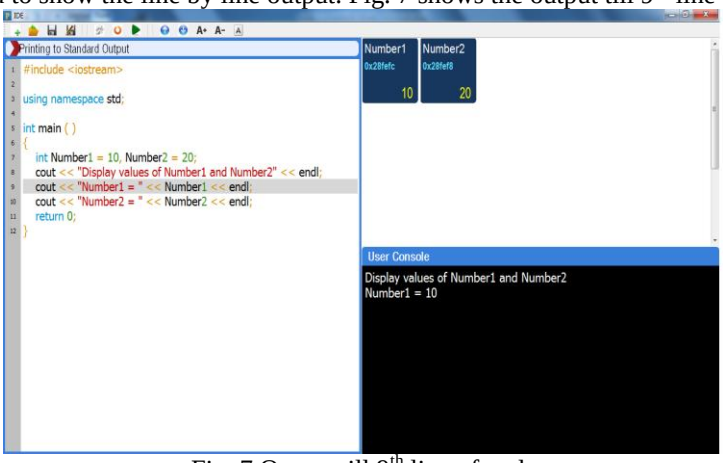

Fig. 7 Output till 9th line of code

Fig. 8 shows the output till 10th line of code.

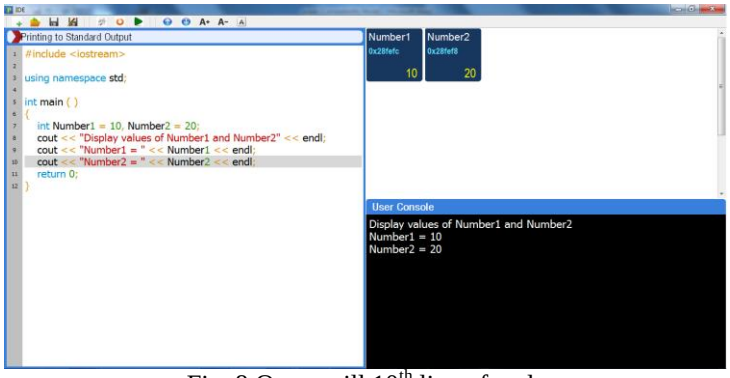

Fig. 8 Output till 10th line of code

III. METRICS FOR VALIDATION

Time and Space complexity have been considered as metrics for validation. Time complexity can be calculated by the amount of time taken by the program to run whereas Space complexity can be calculated by the amount of storage space required by the program. Time and Space complexity are shown with the help of Fig. 9.

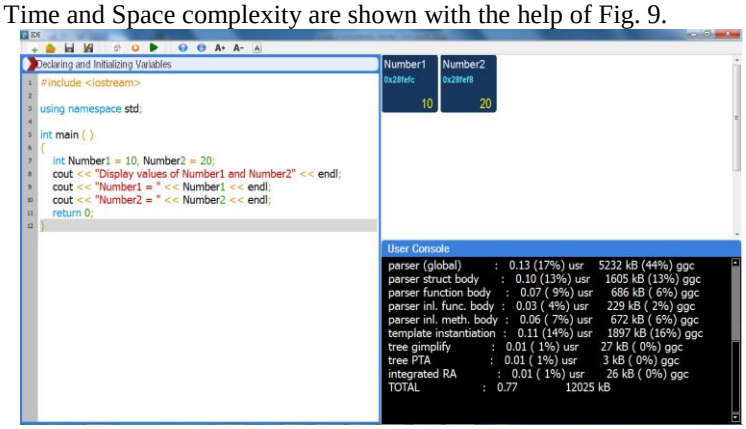

Fig. 9 Time and Space complexity

IV. FEATURES OF IDE

IDE includes four features 1 Flexibility and Extensibility- The IDE is flexible and extensible as we can add, remove and modify functionality without damaging the system. 2 Performance- the Response time of the IDE does not increase even if load increases. So the IDE is scalable. 3 Simplicity and Fast Learn ability- The IDE is simple to use, learn and adapt. 4 Platform Compatibility and Portability- the IDE can run on different platforms like Windows and Linux simply by installing all dynamic link libraries (dll) which are mandatory for this software.

V. CONCLUSION AND FUTURE WORK

There are two key techniques data prefetching and complexity parameter are used in the development of the IDE. With the help of data prefeching, we are able to allocate address as well as value of each variable of the program and it also help in displaying the line by line output of a program. Use of plug-in technology makes the software flexible. The IDE have the ability to develop high-end as well as low-end programs quickly. It helps the beginners to fastly grasp the basic

grammar of the language. It reduces the cost of programmer cultivating and shorten the time of technological learning. With further studies and efficient programming, the complexity can be reduced, more functionality can be added, design can be modified and improved and more plug-ins can be added.

REFERENCES

[1] R.B. Kline, and A. Seffah, "Evaluation of integrated software development environments: Challenges and results from three empirical studies," *International Journal of Human-Computer Studies*, pp. 607–627, July, 2005.

[2] Peng Gao, Lan-Fen Lin, Ming Cai, and Jin-Xiang Dong, "Study of Integrated Product Modeling in Cooperative Development Environment," *The 8th International Conference on Computer Supported Cooperative Work in Design Proceedings*, 2003, pp. 250–254.

[3] Ying-Kui Gu, and Hong-Zhong Huang, "An Integrated Product and Process Development Model Supporting Entire Life Cycle," *The 8th International Conference on Computer Supported Cooperative Work in Design Proceedings*, 2003, pp. 558–562.

[4] Hu Yanju, "Design and Development of C Language Integrated Development Environment with Data Accessing," *International Forum on Information Technology and Applications*, pp. 46–48, 2010.

[5] (2014) Using Layouts in Qt Designer. [Online]. Available: http:// qt-project.org/doc/qt-4.8/designer-layouts.html

[6] (2014) QFrame Class Reference. [Online]. Available: http://qt-project.org/doc/qt-4.8/qframe.html#details

[7] (2014) QPlainTextEdit Class Reference. [Online]. Available: http://qt-project.org/doc/qt-4.8/qplaintextedit.html

[8] (2014) QLabel Class Reference. [Online]. Available: http://qt-project.org/doc/qt-4.8/qlabel.html#details

[9] (2014) Graphics View Framework. [Online]. Available: http://qt-project.org/doc/qt-4.8/graphicsview.html

[10] (2014) QScrollBar Class Reference. [Online]. Available: http://qt-project.org/doc/qt-5/QScrollBar.html#details

[11] Hossein Tajalli, and Nenad Medvidovi´c, "A Reference Architecture for Integrated Development and Run-Time Environments," *IEEE*, pp. 19–24, 2010.

[12] (2014) Syntax Highlighter Example. [Online]. Available:http://qt-project.org/doc/qt-4.8/richtext-syntaxhighlighter.html

[13] (2014) Code Editor Example. [Online]. Available:http://qt-project.org/doc/qt-4.8/widgets-codeeditor.html

[14] Per Bothner, "A Gcc-based Java Implementation," *IEEE*, pp. 174–178, 1997.

[15] R.P.J. Pinkers, P.M.W. Knijnenburg, M. Haneda, and H.A.G. Wijshoff, "Statistical Selection of Compiler Options," *Proceedings of the The IEEE Computer Society's 12th Annual International Symposium on Modeling, Analysis, and Simulation of Computer and Telecommunications Systems (MASCOTS'04)*, 2004.

[16] Shih-Hao Hung, Chia-Heng Tu, Huang-Sen Lin, and Chi-Meng Chen, "An Automatic Compiler Optimizations Selection Framework for Embedded Applications," *International Conference on Embedded Software and Systems*, 2009, pp. 381–387.

[17] (2014) MinGW. [Online]. Available: http://www.mingw.org/wiki/MinGW.

[18] (2014) Standard Input Definition. [Online]. Available: http:// www.linfo.org/standard_input.html

[19] (2014) Standard Output Definition. [Online]. Available: http://www.linfo.org/standard_output.html

[20] Surendra Byna, and Yong Chen, Xian-He Sun, "A Taxonomy of Data Prefetching Mechanisms".

