ZinjaI: An Integrated Development Environment for a first programming course with C++

Pablo Novara[1], Horacio Loyarte[1]
[1]Universidad Nacional del Litoral , Facultad de Ingeniería y Cs Hídricas
Departamento de Informática. Santa Fe, Argentina

Abstract. Most of the students in Argentinian universities tends to experience huge adaptation problem over their first year. It is the main cause for very high indexes of abandonment. In the case of computing/informatics systems careers, in the first programming course, the students must learn a series of concepts related to computing algorithms abstraction, programming language syntax and the real implementation of programs using C++. It is a known fact that this is a cryptic language for the beginner programmer, and usually the very complex Integrated Development Environments (IDE) existing today are not designed to solve this particular issue. Instead, the software seem to be an additional handicap. ZinjaI is a new IDE for writing C++ programs developed with student's needs in mind, with powerful features for making design, edition, debugging and logic tracing of programs simpler tasks. The utilization of this tool in several first year cohorts seems to make a significant improvement for the learning process.

Keywords: Integrated Development Environment, programming teaching, C++.

1 Introduction

Argentinian universities have minimal admission requirements for degree careers and the students pay no fee for their studies. In addition, middle school is going through a crisis. All these factors produce that meaningful number of applicants for coursing a degree career have serious difficulties for successfully finish the first year subjects of its curriculum. These difficulties are increased in the engineering careers, like the case of study (*Ingeniería Informática*, in *Universidad Nacional del Litoral,* Santa Fe, Argentina).

[1] The National Universities are public and free in Argentina.

This career has an initial enrollment of about 300 students, and the drop out in the first year is approximately 50%. Through surveys and assessment process analysis it was possible to detect the subject *Fundamentos de Programación* (Programming Fundamentals) as one of the most difficult ones for students. This subject develops computational algorithm concepts and the students have to solve problems creating computer programs using the standard language ANSI C++.

It was observed that professional C++ programming environment in addition to some cryptic characteristics of the language, tends to confuse and slow down the learning process of programming concepts for inexpert students. The language syntax, the programming environment, the error messages, the English language (the students are Spanish speakers), etc., constitute additional obstacles to basic difficulty when learning the concepts, the logic and basic structures of computational algorithmic: the main objective of the subject.

As a result of this analysis, it was proposed the development of new IDE, ZinajI[1], aimed to learning/teaching needs, with features for facilitating edition, debugging and testing of C++ programs, and contributing in defective way to learning programming basis in general and C++ language in particular.

2 IDE for teaching of programming

The professional IDEs provide important features for accelerating production of complex code to developers. It is common to find in most of them: automatic indentation, syntax highlighting, integrated debugging features for facilitating erroneous logic detection, trace and breakpoints for code analysis, compilation and execution a through menu commands and many other characteristics.

Besides these features for developing programs, in the learning processes, an introductory course of C++ programming requires other characteristics that clash with professional IDE design[2,3]:

- The programming interfaces must be clean, simple and intuitive. The professional IDEs have generally complex interfaces flooded with tools and commands that the student will not take advantage. These kinds of interfaces constitutes an obstacle and distraction for learning.

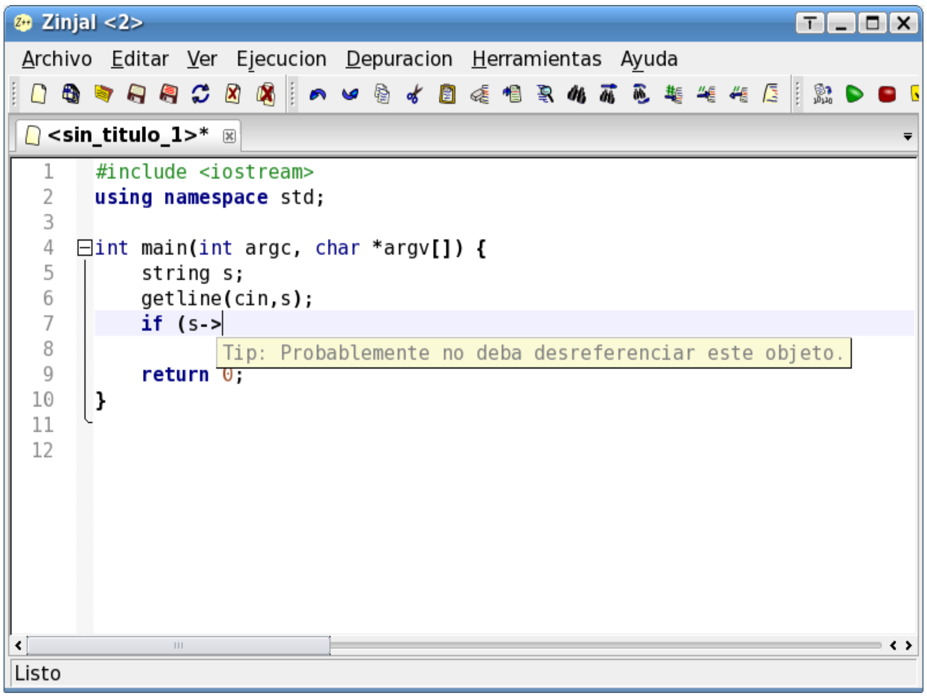
- The setup must be simple and the software installed must have a minimal resource requirement for running in obsolete PCs. It cannot demand high hardware requirements to students who are starting to learn how to program.

- The IDE must provide to the user several levels of helps and assistance in order to improve and smooth the learning process (i.e.: early error detection). That means that the IDE must selects the necessary information for each context and avoid another distracting information (i.e.: compilation parameters, idiomatic barriers, etc).

The powerful features of professional IDEs demand the presence of many command menus and other elements. Most of them will never be used by a beginner programmer and only lead to confusion.

3 Principal features of ZinjaI

ZinjaI was developed having all the topics introduced in the previous section in mind in order to provide a more suitable environment for students. Some its main features are:



```
#include <iostream>
using namespace std;

int main(int argc, char *argv[]) {
    string s;
    getline(cin,s);
    if (s->
    return 0;
}
```

Tip: Probablemente no deba desreferenciar este objeto.

Fig. 1. ZinjaI 's basic interface. It shows emerging message wich warns the user about potential errors detected by the auto-completion system.

- Easy distribution: it is a free and open source software. The whole system (IDE, compiler, debugger, etc.) is deployed in an easy setup package (from 8 to 35 Mb, depending on the platform and version). The software is also prepared to run without any installation process at all.

- Portability: the system can run in both Microsoft Windows (from Windows 98 to latest official release, Windows Vista) and GNU/Linux (in any modern distribution), adapting projects between platforms in a transparent way for the user.

- The initial interface is very simple and clearly intuitive. By each edition tab the system proposes the initial code of a C++ program, so the user can start writing his solution right inside the main function (like shows Fig. 1) through a set of predefined templates or through the new file wizard, and test his program with a single click. So, the system allows the rapid development of C++ programs without need to create, configure and customize projects.

- It has several edition facilities, like syntax highlight, intelligent and automatic indentation, advanced search and replacement, folding and expansion of logical code blocks and some special commands for C++ like automatic header file directive inclusion, management of source's comments, context sensitive auto-complete system, emerging help for function, etc.

- ZinjaI presents a complete help system (IDE documentation, tutorials, advanced features, etc) and an integrated Spanish quick help about standard C++ language.

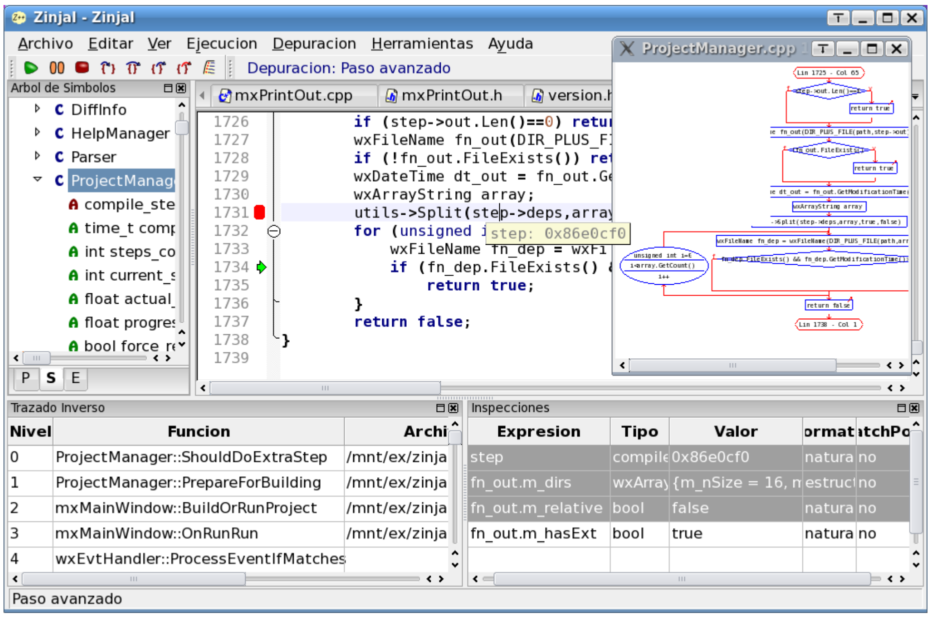
- The system parses and improves compiler output: errors and warnings are organized into a tree shape, restructuring some lines o discarding others, that result in an easier reading and interpretation.

- It also has a Project mode for management of multiple advanced execution and compilation profiles. The fact that new ZinjaI's releases are developed with the old ones show its capability for complex projects handling.

- Debugging system includes inspections management, hierarchical gdb objects exploration, breakpoints (basic breakpoints, conditional breakpoints and watchpoints with full scopes awareness), backtracing, step by step execution, especial table layouts for classes, vectors and matrices, with parsing and reformulation of debugger expressions in order to improve the quality of information presented.

- Very specific student aimed features such as generating and visualizing flow diagrams for selected pieces of code (Fig.2), sharing source and other text files through a LAN network for facilitating the teacher's job, automatic class hierarchy representation, etc.

- Finally, it integrates external tools without adding complexity to basic interface and most common tasks. In addition to student aimed tools, there is a set of advanced components for demanding users: documentation generation through Doxygen, visual interface design through wxFormsBuilder, profiled execution through gprof, source and text files comparison and merging, building scripts generation, etc.



```
        if (step->out.Len()==0) retu
        wxFileName fn_out(DIR_PLUS_F
        if (!fn_out.FileExists()) ret
        wxDateTime dt_out = fn_out.Ge
        wxArrayString array;
        utils->Split(step->deps,array
        for (unsigned i step: 0x86e0cf0
            wxFileName fn_dep = wxFi
            if (fn_dep.FileExists() &
                return true;
        }
        return false;
}
```

Nivel	Funcion	Archi
0	ProjectManager::ShouldDoExtraStep	/mnt/ex/zinja
1	ProjectManager::PrepareForBuilding	/mnt/ex/zinja
2	mxMainWindow::BuildOrRunProject	/mnt/ex/zinja
3	mxMainWindow::OnRunRun	/mnt/ex/zinja
4	wxEvtHandler::ProcessEventIfMatches	

Expresion	Tipo	Valor	ormat	atchPo
step	compile	0x86e0cf0	natura	no
fn_out.m_dirs	wxArray	{m_nSize = 16, m	estruct	no
fn_out.m_relative	bool	false	natura	no
fn_out.m_hasExt	bool	true	natura	no

Fig. 2. Project mode interface in a debugging session. It shows some additional tools in panels, and flow chart representation of the analyzed piece of code.

The help system of ZinjaI describes in great detail all of its features. Of course, a beginner student in first contacts with ZinjaI can work without knowing the advanced ones. The student can incorporate these features while getting fluency in design and development of C++ programming for solving problems.

4 Development of ZinjaI

The software has been built employing a set of completely free and portable tools and libraries. Development was done on a GNU/Linux platform using the standard ANSI/ISO C++ programming language, object oriented programming paradigm, and wxWidgets library for presenting visual components. For compiling and debugging ZinjaI relies on GNU tools (GCC and GDB). Also, some code taken from other free projects was adapted to implement some features (i.e.: the parser that ZinjaI internally uses was taken from the RedHat's Source Navigator project). The main reasons for choosing wxWidgets over other powerful alternatives such as QT, GTK+, FLTK, etc. includes: its object oriented interface, its very high portability and its deep integration with the host operative system, and the fact that it provides a whole framework that also simplify process management with input/output redirection, sockets and other networking components, files and string manipulation, and more.

5 Impact and results in classroom application of ZinjaI

To test the tools acceptance level and main aspects to guide its development an survey was performed. Then, to verify its influence in the educational process the survey was applied in two parallel courses with a significant number of students: one of them (in FICH-UNL[2], where the subject is called *Fundamentos de Programación*, Fundamental Programming, 91 students) was employing the proposed development; the other one (in FI-UNER[3], where the subject is called *Computación I*, 83 students) continued in the traditional teaching mode and without the tool. The comparison gets an special relevance due to:

- very similar topics were taught in both classes
- same teacher in charge of them
- similar number of students
- both courses are in the first year of their respective careers

Analyzing the collected information, several observations can be done:

1. The tool has achieved an important acceptance level in its first intervention: the very first ZinjaI's version was deployed in the middle of course time and more than 75% of the students has adopted it as his main working platform. It must be said that the every student can choose whatever software he wants to carry on the practices, considering that the university labs provide several free and commercial alternatives. Asking for the reasons to the students that choose not to work in ZinjaI the most frequent one is that they where already very familiar with other specific software.

- The tool is user friendly: when the students where interrogated about their reasons to work with it, the main answers where the teacher's recommendation (53%), the ease of use (40%) and the fact that its interface and help is in their home language (36%). In average, the grade of additional difficulty perceived by the students working with ZinjaI when they start programming and change from pseudocode to C++ is very similar. However, when they were asked about the comfort and help level that the software provides the difference is bigger in benefit of ZinjaI (about 3 points in a 1-10 scale for both aspects).

- Some of the strongest system feature where not explicitly appreciated by the students: between the main reasons for choosing that software, “the amount of different features” (considering the specific ones aimed to classroom work) or “low requirements” (as an example, it only takes about 11MB in memory when running in its initial mode) has showed low percentages (around 14% each).

2 FICH-UNL: Facultad de Ingeniería y Ciencias Hídricas, Universidad Nacional del Litoral. Santa Fe, Argentina.

3 FI-UNER: Facultad de Ingeniería, Universidad Nacional de Entre Ríos. Entre Ríos, Argentina.

- Students believes that ZinjaI has a positive impact in the learning process: in contrast to the results exposed in the previous two items, when the students where asked about how they saw the software influence, more than 72% said that the tool contributed to the progress they made, 22% said that it only let them work faster and in a more comfortable way but without a real influence in their academic performance, and only 7% expressed that there was no difference when comparing to other tools and less than 2% that it has a negative impact. Even if those answers are loaded with a high level of subjectivity and very conditioned by the students lack of experience, the big number of individuals in the samples provides some extra credibility to those numbers.

Looking at these results it can be observed that the introduction of the new software into the learning process has made a positive difference for the student's experience. However, the original main design feature was supposed to be a new debugging system conceived to create in the students the habit of taking advantage of debugging as a process not only for finding and fixing their own bugs, but also for analyzing right programs in order to investigate their behaviors and get a better understanding of many theoretical and practical topics introduced in the course [4-9]. In the presented work, the students only have had access to a limited and basic debugging system. This feature was still under heavy development, and that's one reason why it's impact is expected to be actually bigger with the new versions. It also must be said that in those courses there was another tool being tested that had significant influence in the comparison (PseInt[10,11], a pseudo-code interpreter employed in the four first weeks to introduce the most basic and general logical aspects before getting contact with a real programming language), reason why the student's qualifications can't be taken as a direct indicator of how ZinjaI affects the final results.

6 Conclusions and further work

The proposed development has showed a positive impact in the learning process of FICH-UNL students. This fact is supported by partial results extracted from surveys analysis and comparisons between the two selected group of students. The feedback level from academic community (both teachers and students) is determinant and essential to lead the development of new features. The software presented is now in a noticeable more mature and stable state comparing to the releases used for this study. The authors pretend to continue evaluating the impact in the following cohort and improve the software achieving a better integration between the tools usage and the learning/teaching experience.

References

1. ZinjaI: Integrated development environment. Available at http://zinjai.sourceforge.net/
2. Reis Charles, Cartwright Robert: *Tamimng a Professional IDE for the Classrroom*. SIGCSE’04, March 3–7, 2004, Norfolk, Virginia, USA (2004)
3. Moroni, N., “Entornos Para el Aprendizaje de la Programación”
4. Valles Miguel: *Técnicas cualitativas de investigación social*. Editorial Síntesis, Madrid, España (2000)
5. Cross, James H. et. al. “Using the Debugger as an Integral Part of Teaching CS1”, 32nd ASEE/IEEE Frotiers in education Conference, Noviembre 2002.
6. Ko, A. J., “Preserving Non-Programmers’ Motivation with Error-Prevention and Debugging Support Tools”. 2003.
7. Chmiel, R. y Loui, M. C., “An Integrated Approach to Instruction in Debugging Computer Programs”. 33rd ASEE/IEEE Frotiers in education Conference, Noviembre 2003
8. Nagvajara, P. y Taskin, B., “Design-For-Debug: A Vital Aspect in Education”. Internacional Conference on Microelectronic Systems Education, 2007.
9. Gallego, C. M. et. al., “Depuración Estructural: Acercando la teoría a la práctica de la Programación”.
10. Loyarte Horacio. y Novara Pablo: “Desarrollo de un Intérprete de Pseudocódigo para la Enseñanza de Algorítmica Computacional”. I Congreso de Tecnología en Educación y Educación en Tecnología. TE&ET. La Plata, Argentina (2006)
11. PseInt: Spanish pseudocode interpreter. Available at http://pseint.sourceforge.net/