WIPE (Web Integrated Programming Environment): Design Principles and Architecture

Vassilios Efopoulos[1], Georgios Evangelidis[1], and Vassilios Dagdilelis[2]

[1] Department of Applied Informatics, University of Macedonia
[2] Department of Educational and Social Policy
University of Macedonia, 54006 Thessaloniki, Greece
Email: {efop, gevan,dagdil}@uom.gr

Abstract The scope of the present study is the description of WIPE (Web Integrated Programming Environment), a web-based environment used for teaching programming for beginners. The WIPE system (alpha version) is accessible on the Internet via a web browser (`http://macedonia.uom.gr/~efop`) and consists of four major components: (1) an e-learning environment, (2) a content management system based on an open source environment, (3) supporting tools (among them a specially featured web-based compiler based on and further expanding the X-Compiler environment of the educational software DELYS), and (4) a specially designed database that is used for storing several types of data.

1 Introduction

WIPE (Web-based Integrated Programming Environment) is a complete web-based database driven environment that serves for the teaching of the basic principles of programming. Such environments that support the teaching of information technology have attracted the interest of the research community. Thus, a significant number of environments that support, in particular, the teaching of programming to beginners, have appeared in the literature. Further, many of these environments focus their attention on the comprehension difficulties novice programmers meet and the mistakes that they usually make. Below, we classify the various environments that have been suggested:

- Programming microworlds that assist the first steps of the students by providing a familiar environment, such as Karel the Robot [13] and Karel Genie [10].
- Compilers supported by programming tools that facilitate the authoring of programs (debuggers, watchers - listeners), such as Thetis C [9].
- Visual programming environments that use alternative ways to represent basic programming operations (e.g. flowcharts), such as BACCII [5].
- Environments that allow a step-by-step execution of the program code and visualization of the used data structures, such as Dynalab [2].

In the last years, the rapid growth of Internet brought about suggestions for programming environments to be used in the World Wide Web. The most significant attempt at this direction is implemented in the University of Montana by the team

that created Dynalab. The outcome of this effort is the WebLab project, which is basically the Internet version of Dynalab [3] [4]. WebLab's first version supports a subset of Java.

The programming environment we propose and describe in this paper is based on and further expands the compiler environment of the educational software DELYS [6] that was developed under the framework of the pilot project NAFSIKA-ODYSSEIA supervised by the Greek Ministry of Education. The experience that the research group acquired by the use of DELYS in secondary education was a crucial factor that led to the decision to expand it for use over the web.

Among the features of the suggested environment are:

- A complete e-learning environment,
- A Content Management System based on an open source environment (Postnuke [15]) licensed under the GNU General Public License.
- A collection of learning tools, such as, a web-based compiler of a programming language which is translated into a pseudo assembly language,
- A database that is used for storing several types of data (input by teachers, outcomes of the student homeworks, succesive student solutions to exercises etc.).

WIPE components are mainly implemented in C, PHP [14] and Macromedia Director [11] (Shockwave output) for being deployable on and compatible with heterogeneous systems.

In Section 2 we present a detailed description of the proposed system. We describe the learning environment, the web-based compiler and the CMS website. In Section 3 we briefly discuss the tools we used for implementing the system, and finally, we conclude in Section 4.

2 Description of the WIPE System

WIPE operates via the Internet or a local Intranet and is accessible through a web browser. The software is installed on a network server (Intranet or Internet). In the following subsections we will describe the functionality of the major components of WIPE, namely, the e-learning environment, called ELearner, the web-based compiler, called WIPE-compiler, and the CMS website.

2.1 ELearner

ELearner is the main module of the WIPE system. It is an e-learning environment that could even operate as a separate tool. It consists of a User and Group Management and a Learning Material Management module. ELearner is accessible through a web browser in the following URL `http://macedonia.uom.gr/~efop`.

User and Group Management. The system supports four different access modes: (a) guest user, (b) teacher, (c) student (defined by a teacher), and (d) system administrator.

(a) Guest user access. Guest users (i.e., anybody browsing the WWW) self-register and can access the system via a web browser, at a specific URL. Such users have the right to use only a fraction of the available features. In particular, they are allowed to view (1) the content management system (the web site, the news and the teacher's announcements) (2) public exercises created by the teachers (3) certain supporting tools (such as the WIPE Compiler or the Virtual Scale). They can develop and run programs and inspect their output. In this scenario, user actions are not recorded in the database.

(b) Teacher access. Users that request to access the system as teachers have to fill-out a self-registration form that is available via a link in the main page of the website. This form is automatically delivered to the system administrator. The administrator examines the request, approves it and sends an email with a teacher password to the applicant. Teachers who access the system using this password have the following options:

- They can define their lessons and exercises (described in the next section), the user groups and subgroups they teach and also the school department they belong to.
- They can register the students for each group or subgroup. For each student they register, they have to specify a username and a password for entering the system.
- They can define homework exercises for a specified group. A homework exercise is the basic element of WIPE and is administrated by a teacher. They can also define the students that will be allowed to participate in it by choosing specific groups, subgroups or individuals. For each exercise, they must enter the following information: description, objectives, closing date, help, resources (e.g. links) and participants.
- They can define workgroups, that is, teams of students that have to work together in order to complete an exercise. Each workgroup shares a specific folder (storage area in the server) that is accessed only by its members. Workgroups are used when teachers want to assign large scale programming projects to specific students.
- They can define public homework exercises, that is exercises that are accessible by all users (i.e. teachers, students, and guests).
- They can monitor the performance and progress of their students.

In summary, a teacher is the administrator for the lessons, exercises, subgroups, workgroups, and students that he/she registers. A teacher submits and stores material (such as exercises, resources etc.) and then makes it available to the students whenever he/she wishes.

(c) Student access. The most important, and central to the proposed environment, type of users is students. Users that enter an appropriate username and password and login to the system as students, have the right to use all the supporting tools their teacher has decided to provide. For example, when using the WIPE compiler, students can:

- explore the programming environment and write their own programs that are recorded (i.e., stored in the database) each time they are compiled,

- execute their programs in different ways (such as continuous execution or step-by-step execution of the source or assembly code) with simultaneous display of the user and system variable values,
- ask the system to check the correctness of their programs,
- solve programming exercises that are defined by their teacher, and
- submit their solutions to the exercises to their teacher.

As soon as a teacher defines an exercise, the system automatically generates a personal folder for each participating student. It also generates a supporting folder (library) for the exercise that contains all the supplementary material. We wish to point out that each time a student compiles a program the system records in the database the name of the student, the time, a copy of the source code, and the errors that occurred.

(d) System administrator access. The system administrator is a user with unlimited access rights that can add, modify, and delete teachers, students and guests.

Learning Material Management. The learning material is organized in lessons with a given structure of folders and files. A lesson is comprised of exercises, a library of learning resources to be used as supporting material by the students, and private storage area for the documents produced by each student. An exercise is constructed for a group of students by a teacher. It has a description including context and goals.

When a lesson is created, the following folder structure appears in the server catalogue:

```
Lesson
    Exercises
        One folder per Student & Workgroup (Upload Area)
            One folder per Exercise
    Supporting Material
    Storage Area
        One folder per Student & Workgroup (Private Storage Area)
```

The exercises folder contains the solutions to exercises submitted by students or student groups. The supporting material folder includes files that the teacher uploads to help students accomblish their homework. Finally, the storage area can be used by students of student groups as a permanent and private area to keep their files.

The only users that can modify the contents of the above folders are (1) the administrator and (2) the teacher who created the lesson. Students can modify their personal folder and the workgroup folders they belong to.

2.2 WIPE Compiler

The WIPE Compiler is a web-based programming environment and it is one of the supporting tools of the WIPE System. It is a specially featured web compiler that, in its current version, translates the source code of a Pascal-based programming language into a pseudo assembly code.

The WIPE Compiler allows students to develop their own source language programs, visualize their execution, and test their correctness. The GUI of WIPE is embedded in the environment of the web browser and consists of the following elements:

- A source code window that is used for composing and editing the program (source code).
- An assembly code window that displays the compilation or the source code into assembly.
- An output window that displays the outputs of the program's execution.
- A message window that displays annotations, hints, exercise descriptions and messages during the compilation procedure.
- A register window that displays the values of the vitrual machine registers (in our case two registers).
- Toolbars that include the most common commands (compile, run, cancel, copy, paste, etc).

The WIPE compiler environment and its philosophy of functioning incorporate a number of features with didactic value:

- Students can monitor all the intermediate states during the execution of their program: compilation, assembly code, registers, and intermediate values of all user and system variables. Furthermore, they can edit directly the assembly code and then execute it.
- Students can ask the compiler to verify the correctness of their code. This is done with the use of "algebraic assertions", a type of black box testing. This feature is explained in [16].
- The compiler provides students with many explanatory messages that are as accurate as possible and can help novice programmers to debug and generally improve their programs.

In Figure 1, one can see the actual programming environment of the WIPE compiler as seen through a web browser. The WIPE compiler is accessible via the WWW at `http://macedonia.uom.gr/~efop`.

The Programming Language of the WIPE Compiler. Currently, the WIPE Compiler supports source code programs written in a subset of Pascal. Pascal is a commonly used programming language in the secondary and tertiary education curricula.

The WIPE Compiler language supports: identifiers, assignment, I/O statements (write, read), conditional statements (if...then, if ... then ... else), loop statements (while ... do, repeat ... until), compound statements (begin ... end), comments, arithmetic and boolean expressions.

The Assembly of WIPE Compiler. The assembly language used in the WIPE Compiler is a pseudo-assembly that runs on a virtual machine with two registers. The pseudo-assembly includes:

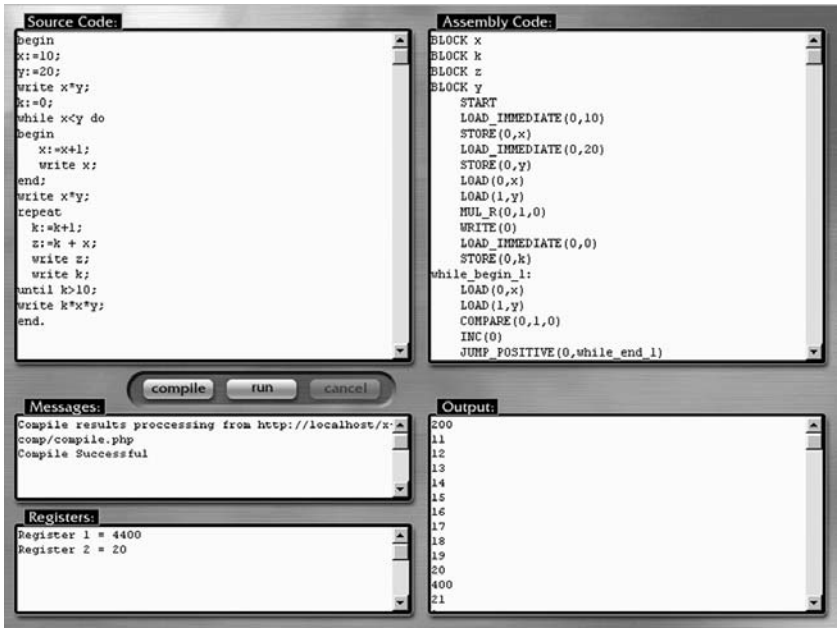

Figure 1. The WIPE Compiler programming environment.

```
Move & Store instructions (LOAD, LOAD_IMMEDIATE, STORE, MOV_R)
Compare instructions (COMPARE)
Arithmetic instructions (ADD_R,SUB_R, MUL_R, DIV_R, INC, DEC, NEG)
Jump Instructions (JUMP, JUMP_POSITIVE, JUMP_NEGATIVE, JUMP_ZERO)
Input/Output Instructions (READ, WRITE)
```

The WIPE system can allow for the addition of any desired language (C, Java, etc.) and the simultaneous support of many different compilers. All one has to do is develop the appropriate front-ends for each language. All front-ends should compile source language statements into the WIPE Compiler pseudo-assembly.

The Compilation Procedure. In Figure 2 we demonstrate the procedure that is followed during the compilation of student exercises:

- Students connect to the system and select a lesson. Then they select one of the available exercises. The teacher must have assigned the WIPE Compiler as a supporting tool for that exercise. Students must load the WIPE Compiler environment by visiting the appropriate link and use the "source code" window to enter their code. At any time they can compile their program, in which case the source code is sent to the web-server.
- The web-server stores the code into the DBMS and also forwards it to the WIPE Compiler.

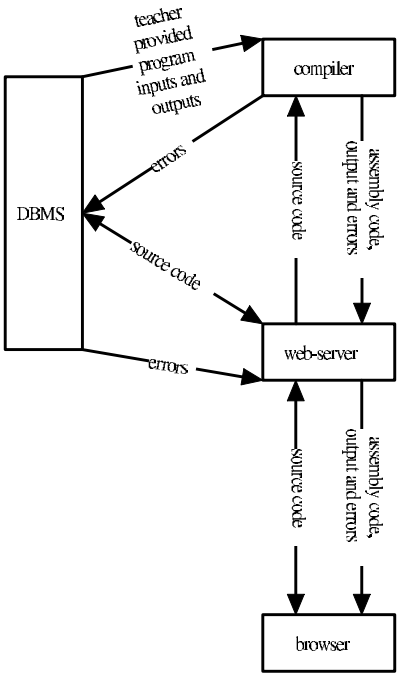

Figure 2. Programming environment architecture.

- The source code is compiled and in the case of (a) success, the assembly code is sent back to the student's web browser via the web-server, (b) failure, the encountered errors are stored in the DBMS and are also sent back to the student's web browser via the web-server.
- If the compilation is successful the student can select the run command in which case the compiled program is executed on the server and the program's output is transferred to the student's web browser.

2.3 The CMS Web Site

The CMS web site that the WIPE System incorporates enables administrators and teachers to create, publish, and manage content such as news, announcements, specific articles, etc. This system enables administrators to maintain a Web site for their school department. They have a unique username/password that provides total control over the content, access rights, and look & feel of the web site. The system has easy-to-use tools allowing teachers (content providers) to work within a structured environment and rapidly deliver content such as news, articles, job boards, web links, dynamic headlines, frequently asked questions, chat and file download areas. Teachers can write their own articles in a word processor that the system includes and publish them directly on the web via an Internet browser.

3 Implementation

The WIPE server-side system components are mainly implemented in C, Lex, Yacc [1] and PHP for being deployable on and compatible with heterogeneous systems (Windows, Unix, MacOS environments). The WIPE system clients are actually web browsers with Shockwave plug-in support since the WIPE Compiler environment has been implemented using Macromedia Director (Shockwave output). Currently, only Windows and MacOS based browsers support such a feature.

The Elearner component, that comprises the main gateway to the WIPE system has been implemented in PHP. The Content Management System is implemented in PHP and MySQL [12] and is based on the open source CMS environment Postnuke.

4 Conclusion

As we have already mentioned researchers have made several significant efforts to develop educational environments for helping new programmers and for supporting the teaching of the basic principles of programming. These environments were based on specific research that revealed misconceptions and comprehension difficulties of students [8]. In the next paragraphs, we sum up the basic features that WIPE incorporates and, in our opinion, are most important for supporting the teaching and learning of programming.

- The ability to execute a program step by step accompanied by simultaneous observation of the values of user and system variables and the ability to verify and disprove programs helps new programmers understand the way programs are executed and the internal procedures that take place when the computer runs a program. Several studies have shown that these subjects are not easily explicable to novices [7] [8].
- The message window will make debugging an easier task by displaying annotations and hints. It has been observed that error messages from most programming tools are not user friendly at all and cause confusion to students [9].
- The fact that the system records all the attempts that a student makes in order to develop a program could be very useful to teachers, since it will provide them with valuable information about the difficulties that students experience. In this way, teachers can adapt their teaching to the students' needs.
- The system also provides the teacher with statistical information about the exercises and the student performance. This way the teacher has the opportunity to evaluate the efficiency of a given exercise and probably revise it, if this seems to be necessary.
- Last but not least, all the results of this project will be available through the Internet to all teachers and researchers. We hope that they will provide them with useful information about the way students conceive the principles of programming. The WIPE System will be evaluated in a large-scale experiment involving schools and universities throughout Greece. Outcomes of these experiments will be used to improve the WIPE system, resulting in the second prototype (beta version).

References

1. Aho A.V., Sethi R., Ullman J.D.: *Compilers: Principles, Techniques, Tools.* Addison-Wesley (1988)
2. Birch M., Boroni C., Goosey F., Patton S., Poole D., Pratt C., Ross R.: DYNALAB: a Dynamic Computer Science Laboratory Infrastructure Featuring Program Animation. *Proceedings 26th ACM SIGCSE Technical Symposium on Computer Science Education.* (1995) 29-33
3. Boroni C., Goosey F., Grinder M., Ross R.: A Paradigm Shift! The Internet, the Web, Browsers, Java, and the Future of Computer Science Education. *Proceedings 29th ACM SIGCSE Technical Symposium on Computer Science Education*, (1998) 145-149
4. Boroni C., Goosey F., Grinder M., Ross R., Wissenbach P.: WebLab! A Universal and Interactive Teaching, Learning, and Laboratory Environment for the World Wide Web. *Proceedings 28th ACM SIGCSE Technical Symposium on Computer Science Education*, (1997) 199-203
5. Calloni B., Bagert B.: Iconic Programming in BACCII vs. Textual Programming: which is a Better Learning Environment? *Proceedings 25th ACM SIGCSE Technical Symposium on Computer Science Education*,(1994) 188-192
6. Dagdilelis V., Evangelidis G., Satratzemi M., Efopoulos V., Zagouras C.: DELYS: a Novel Microworld-based Educational Software for Teaching Computer Science Subjects. *Computers & Education*, 40:307-325 (2003)
7. Du Boulay B., O'Shea T., Monk J.: The Black Box Inside the Glass Box: Presenting Computing Concepts to Novices. In: Soloway E., Sprohrer J. (eds.): *Studying the Novice Programmer.* Lawrence Erlbaum Associates, (1989) 431-446
8. Du Boulay B.: Some Difficulties of Learning To Program. In: Soloway E., Sprohrer J. (eds.): *Studying the Novice Programmer.* Lawrence Erlbaum Associates, (1989) 283-300
9. Freund S.N., Roberts E.S.: THETIS: an ANSI C Programming Environment Designed for Introductory Use. *Proceedings 27th ACM SIGCSE Technical Symposium on Computer Science Education*, (1996) 300-304
10. MacGNOME Project. Computer Science Department, Carnegie Mellon Univ., Pittsburgh, PA 15213
11. Macromedia. `http://www.macromedia.com`
12. MySQL. `http://www.mysql.com`
13. Pattis R.E., Roberts J., Stehlik M.: *Karel - the Robot, a Gentle Introduction to the Art of Programming*, 2nd edition, Wiley, New York (1995)
14. PHP: Hypertext Preprocessor. `http://www.php.net`
15. Postnuke. `http://www.postnuke.com`
16. Εφόπουλος, Β., Ευαγγελίδης, Γ., Δαγδιλέλης, Β., Κασκάλης, Θ.: WIPE – Ένα Διαδικτυακό Προγραμματιστικό Περιβάλλον με υποστήριξη Βάσεων Δεδομένων για τη διδασκαλία των αρχών του προγραμματισμού. Πρακτικά 2ου Πανελλήνιου Συνέδριου των Εκπαιδευτικών για τις ΤΠΕ, 'Αξιοποίηση των Τεχνολογιών της Πληροφορίας και της Επικοινωνίας στη Διδακτική Πράξη'. Σύρος 9–11 Μαϊου (2003)