Objects-Early Tools – A Demonstration

Joe Bergin
Computer Science
Pace University
berginf@pace.edu

Kim Bruce
Computer Science
Williams College
kim@cs.williams.edu

Michael Kölling
Mærsk McKinney Møller Institute
University of Southern Denmark
mik@mip.sdu.dk

SUMMARY
Various software tools have been proposed or developed for use in introductory programming courses. Usually, presentation of a new tool at the SIGCSE symposium occurs when a tool is first developed, leaving it to interested observers to identify success or failure of tools over their lifetime.

For teachers, it can be difficult to compare available tools and judge potential application in their courses.

In this session, three tools with an established track record of successful classroom use are presented: Karel J Robot [1], objectdraw [2], and BlueJ [2]. In addition to considering each tool individually, opportunities for combinations of these tools are also discussed.

The focus of this session will be on concrete, hands-on advice that teachers can immediately apply in their classrooms.

Categories and Subject Descriptors
K.3.2 [**Computers & Education**]: Computer & Information Science Education - *Computer Science Education*

General Terms
Human Factors.

Keywords
CS1, object-orientation, objects early, pedagogy, software tools.

1. INTRODUCTION AND OBJECTIVES
The idea of teaching a programming course in an 'objects-early' mode has been proposed and discussed repeatedly in the past. A common observation, however, is that many teachers still find it very difficult to find a good entry point into teaching such a course.

Over the last few years, a considerable number of software tools have been developed and presented at the SIGCSE symposium to help with this task. Reports from users consistently show that use of some tools can greatly ease the task of dealing with some of the overhead that teaching of object orientation imposes on teachers and students, and significantly increase understanding of important concepts by students.

New teachers, however, do not have an easy task in finding and evaluating existing tools, identifying those that are still being maintained, and keeping up with recent changes and developments. Frequent questions appearing on various teaching-related mailing lists or being sent to the authors directly bear witness of these difficulties.

In this session, three tools that have a proven track record of successful classroom use are being presented by their authors. These presentations aim at achieving three things:

- to give teachers who know little or nothing about the tool a quick and general introduction into capabilities and application areas of the tool;
- to give teachers who already know the tool an update on important recent changes in the tool; and
- to provide concrete, immediately useful ideas about improving programming courses through the use of software tools, that teachers can directly benefit from.

The three projects presented are well known instances of three different types of object-oriented software tools: a micro world (Karel J Robot), a development environment (BlueJ) and a class library (objectdraw).

All three have in common that they have been developed specifically for the teaching of object-oriented concepts in introductory programming courses. All three are also well beyond experimental stages, having been widely used for several years.

Presenting the tools together in a single session provides the unique opportunity to not only discuss each of the software tools individually, but to investigate how these software tools may be combined or complemented to increase their impact.

2. KAREL J ROBOT
Karel J Robot is the current version of a microworld that has evolved since about 1980 into an effective means of teaching programming to novices. It has always focused on correct use of the then dominant programming paradigm. Today that is object-oriented programming. Karel thus introduces dynamic polymorphism as early as reasonable in a course (about the third week of CS1). The Karel world is visual, programming is in Java, and the system is provably a Turing Machine, permitting very complex programs to be written with a very reduced Java subset. Problem solving is stressed over language syntax, and instructors are dissuaded from introducing new language features to solve new problems.

The Karel J Robot (KJR) system is distributed as a Java jar file that has the visual system and a set of classes for primitive robots that the student programmer extends to do interesting things in a simple world. In addition to the "obvious" things (described in the manuscript), KJR provides an agile testing framework (JUnit based) so that test first development can be done in KJR. There is also an acceptance testing framework (KarelFixture) permitting

SIGCSE'05, February 23–27, 2005, St. Louis, Missouri, USA.
ACM 1-58113-997-7/05/0002.

the instructor to present exercises as executable specifications interleaved with text descriptions of the exercise requirements.

KJR also has a remote controller interface permitting students to exercise robots manually. This is built with Java reflection, permitting the student's own code to be so exercised easily. Finally an event driven programming API has recently been added to KJR so that something like programming physical robots via feedback can be simulated.

Web site: `http://csis.pace.edu/~bergin/KarelJava2ed/Karel++JavaEdition.html`

3. BLUEJ

BlueJ is an integrated Java development environment specifically designed for introductory teaching that presents a unique front-end to offer an interaction style that is different to other environment available today.

BlueJ was carefully designed with three fundamental goals in mind: *visualization*, *interaction,* and *simplicity*.

The visualization concept aims at making the main abstractions of object orientation visible on screen: objects and classes. This makes it much easier for students to understand and reason about these fundamental concepts, and easier for teachers to talk about them.

Classes are visualized in a UML-like diagram, which gives an immediate overview over students projects and also shows dependencies. Objects are shown when they are instantiated.

The interaction part enables users to directly interact with the conceptual entities in the environment. Classes can be interactively instantiated, and public method can be invoked on objects. This encourages an exploratory approach to understanding object orientation that involves the student much more than traditional environments.

Simplicity is a key goal in BlueJ's interface design. The environment itself should not become an obstacle in the learning of programming principles. Unfortunately, environments designed for professional developers are of a complexity that poses its own challenges, distracting from the learning of programming itself. BlueJ avoids this by presenting tools custom-designed for student use.

Apart from the general object interaction mechanism, BlueJ offers integrated support for a variety of education-related tools, most importantly an easy-to-use debugger, a *javadoc* generation facility and integrated support for JUnit.

BlueJ was first released in 1998, and is now widely used in many institutions all over the world.

Web site: `www.bluej.org`

4. OBJECTDRAW

The objectdraw library was developed by Kim Bruce, Andrea Danyluk, and Tom Murtagh to support teaching Java to novice programmers. It is designed to support an "objects from the beginning" approach to CS 1. It supports truly object-oriented graphics, makes it possible to incorporate event-driven programming techniques from the beginning, and provides support for introducing concurrency quite early in a course.

Graphical objects both serve as excellent examples of objects and provide visual feedback that makes it easier for students to determine the effects of their code and to detect errors in their programs. The graphical objects provided in the objectdraw library also have the advantage that results of the creation and modification of objects appear immediately on the screen without the need to invoke a paint or repaint method.

The programs that students use most are event-driven. Clicking on a link, selecting an item from a menu, or dragging an icon results in program actions. Students are more highly motivated by writing event-driven programs than using traditional text-based I/O. Moreover, because different events can happen at any time, students learn early that event-driven methods can be invoked in many different orders, just as the methods of other objects can be invoked in different orders. Thus students learn from the beginning not to expect a single monolithic ordering of program statements during execution. The objectdraw library contains a class `WindowController` that extends `Japplet` by inserting a drawing canvas in the center of a window and that serves as a listener for mouse actions on the canvas. The syntactic simplifications that result make it easy for students to program in an event-driven style. In many cases, event-driven programming is easier than more traditional styles.

Web site: `cortland.cs.williams.edu/~cs134/eof`

5. PRESENTERS

Joe Bergin is Professor of Computer Science at Pace University. He has been teaching for 32 years and OO programming since 1986. He is the creator of Karel++ and Karel J Robot, based on earlier work of Pattis, Stehlik, and Roberts. He has written widely on OO programming and especially the central role of polymorphic thinking.

Kim Bruce is the Frederick Latimer Wells Professor of Computer Science at Williams College, and is currently a Visiting Professor of Computer Science at the University of California at Santa Cruz. He is the author of *Foundations of Object-Oriented Languages: Types and Semantics*, MIT Press, and is a co-author of the forthcoming CS 1 text, *Java: An eventful approach*, Prentice-Hall.

Michael Kölling is an Associate Professor of Software Engineering at the University of Southern Denmark. He is one of the developers of the BlueJ environment and has published numerous papers on object-oriented and computing education topics and is co-author of a successful Java textbook.

6. REFERENCES

[1] Bergin, J., Stehlik, M., Roberts, J., Pattis, R., Karel J Robot: A Gentle Introduction to the Art of Object-Oriented Programming in Java, Unpublished manuscript. Available on the web at: http://csis.pace.edu/~bergin/KarelJava2ed/Karel++JavaEdition.html

[2] Bruce, K. B., Danyluk, A., and Murtagh, T. A library to support a graphics-based object-first approach to CS 1. In Proceedings of the 2001 ACM SIGCSE Symposium (2001), pp. 6–10..

[3] Kölling, M., Quig, B., Patterson, A. and Rosenberg, J., The BlueJ system and its pedagogy, Journal of Computer Science Education, Special issue on Learning and Teaching Object Technology, Vol 13, No 4, 249-268, Dec 2003.