jGRASP: An Integrated Development Environment with Visualizations for Teaching Java in CS1, CS2, and Beyond

James H. Cross II, Dean Hendrix, and David A. Umphress
Computer Science and Software Engineering, Auburn University
Auburn, AL 36849-5347, cross@eng.auburn.edu, hendrix@eng.auburn.edu, umphrda@auburn.edu

Abstract - jGRASP is an integrated development environment, created specifically to provide automatic generation of visualizations for improving the comprehensibility of the software. These visualizations, which are particularly well suited for CS1 and CS2 students learning Java, include Control Structure Diagrams, UML Class Diagrams, and new dynamic Object Views. The object workbench and integrated debugger facilitate a unique way for students to view objects created by their programs. The workshop will include tutorials and example programs to demonstrate how instructors can improve the learning and programming experience of their students with jGRASP. Instructors are encouraged to bring programs from their own courses to experiment with the visualizations during the workshop. JGRASP is freely available at www.jgrasp.org. The jGRASP software and workshop materials will be provided.

Index Terms – Computer programming, Java, Programming development environment, Software visualization.

Workshop Description

The jGRASP workshop will include tutorials and example programs to demonstrate how instructors can improve the learning and programming experience of their students with jGRASP and the visualizations it produces. Instructors are encouraged to bring programs from their own courses to experiment with the visualizations during the workshop. JGRASP is freely available at www.jgrasp.org.

This workshop will be conducted in computer lab and the jGRASP software and workshop materials will be provided for each participant. The topics to be covered and objectives for each part of the workshop are briefly described below.

- **Getting Started** – Editing, compiling, and running Java programs in jGRASP. In addition, the pedagogical features will be introduced, including the Control Structure Diagram (CSD), generating the CSD, folding your source code, numbering the lines, and stepping through the program in the integrated debugger.
- **Objects First** – Using projects, UML class diagrams, and the Object Workbench in jGRASP. These topics are especially relevant for an objects first or objects early approach to learning Java.
- **Using Projects** – Creating projects, adding files, removing files, generating documentation, and understanding the difference between the active project and other open projects.
- **Using UML Class Diagrams** – Generating the UML class diagram for a project, displaying the members of a class as well as the dependencies between two classes, and navigating to the associated source code.
- **Using the Object Workbench** – Creating objects for the workbench from the classes in the UML diagram, invoking the methods for each of these objects, and displaying the dynamic state of an object by opening an Object Viewer for it.
- **Using the Debugger** – Setting breakpoints and stepping through programs, either by single stepping or auto stepping, and displaying the dynamic state of an object by opening an Object Viewer for it.
- **Using Control Structure Diagrams** – Using and understanding the graphical notations used in the CSD for basic control constructs of modern programming language, including sequence, selection, iteration, exits, and exception handling.

The figure on the following page illustrates the jGRASP development environment.
FIGURE 1. EXAMPLE OF JGRASP DEVELOPMENT ENVIRONMENT.