Chapter 18
Software Tools and Virtual Labs in Online Computer-Science Classes

Vladimir V. Riabov
Rivier College, USA

Bryan J. Higgs
Milford, NH, USA

ABSTRACT
The authors share their experiences in teaching various online computer-science courses (via the Blackboard™ and synchronous web conferencing tools) and using state-of-the-art free-license software tools for conducting online virtual labs and numerous students’ projects. The labs were designed to help students explore modern sophisticated techniques of computer-system analysis and design, programming in C/C++ and Java, data communication in networking systems, system simulation and modeling, image processing, multimedia applications, Web development, and database design and management. All the online courses include lab-based projects that provide students with knowledge, instructions, and hands-on experience, and motivate them in selecting topics for technology overviews and research studies.

INTRODUCTION
Hands-on technology-exploration experience for students is an integral part of the traditional Computer Science Curriculum. It becomes a challenge for an instructor to transfer such experience to students via online instructions (Higgs & Sabin, 2005; Riabov, 2006a; Sabin & Higgs, 2007). First, a student should have access to state-of-the-art free-license software tools that are used both in academia and the industry. Second, a student should have an opportunity to emulate a computer system (in his/her home or office) by following online tutorials or instructor’s lab manuals. Both tasks require the search for adequate software tools and the development of proper tutorials and lab manuals for every computer-science course offered online or as a hybrid one (Higgs & Sabin, 2005; Riabov, 2006a).

This paper contains an overview of the state-of-the-art free-license software tools, tutorials, lab manuals, some lab reports, and related research papers published by Rivier College students who took computer science courses online or in a hybrid format. Examples of lab manuals, students’ project
reports and publications are also available on the instructor’s website (Riabov, 2009). Advantages of using synchronous web conferencing tools in online course teaching and learning are also discussed.

TOOLS AND LABS FOR ONLINE COMPUTER-SCIENCE COURSES

Computer System Analysis and Design

Tools for Implementation of the Unified Modeling Language Approach

Visual Paradigm™ for UML (Community Edition) tool (Visual Paradigm, 2009) was selected for exploring and conducting virtual labs in online courses on software engineering and computer/information system analysis and design. This tool allows students to build various diagrams by using the Unified Modeling Language (UML) approach in object-oriented system design (Dennis, Wixom, & Tegarden, 2009). Online tutorials and lab manuals provide students with the instructions and examples of developing Use-Case, Activity, Class, Object, Collaboration, Sequence, and Package diagrams and State-machine charts that have become a part of students’ projects. The limited number of diagrams could also be built by using ArgoUML™ (2009) and VIOLET™ (2009) Open Source software tools.

Examples of Student’s Projects for Object-Oriented System Design

All the software engineering steps in the development and implementation of the individual capstone project by using the object-oriented design approach are described in detail by David Snogles, a Rivier College graduate student, in his article, “Personal Encrypted Talk - Securing Instant Messaging with a Java Application” published recently in the Rivier College Online Academic Journal (Snogles, 2005). The primary goal of the project was to secure Instant Messaging Communication between two parties on the Internet. The Personal Encrypted Talk (PET) system was designed to operate on Microsoft XP and Microsoft 2000 machines with an active connection to the Internet. Microsoft Visio™ with UML Template and Jude™ UML Community Edition (Version 1.4.3) have been used as UML tools for object-oriented system analysis and design. The State Chart has provided details into the interaction of multiple users with the PET application. It illustrates the exchange of secret keys according to the Diffie-Hellman asymmetric key agreement protocol. The sequence diagram was used to solidify the interaction of the main components of the PET system and its end users. UML diagrams were developed for all eleven classes and integrated into a system-level UML class diagram.

The Unified Modeling Language (UML) approach has been used by John Dion, a Rivier College graduate student, in his project, “Musician Web-service Using Ruby-on-Rails, SOAP, FLEX, and AJAX” recently published in the Rivier College Online Academic Journal (John Dion, 2006b). His system provides musicians with a web service to communicate with other musicians and their fans. The large package exchange is implemented. The Use-Case diagrams and User-Interface State charts have been used in the system analysis and design for this project.

Programming Fundamentals

In programming courses, one issue that commonly arises is whether or not to use an Interactive Development Environment (IDE). It is certainly possible to use only an intelligent editor, combined (usually) with the command line interface for that operating system, programming language compiler and run-time. Suitable intelligent editors typically can be configured (or come preconfigured) for the programming language, providing suitable syntax