TOWARDS USING ONLINE PORTFOLIOS IN COMPUTING COURSES*

Bryan Higgs, Ph.D.**
Associate Professor, Department of Mathematics and Computer Science, Rivier College
and
Mihaela Sabin, Ph.D.***
Associate Professor, Department of Mathematics and Computer Science, Rivier College

Keywords: *Online portfolios, proprietary vs. open-source web authoring*

Abstract

The direct experience we had with teaching a summer pre-college computing course in which we adopted an online portfolio approach has led us to consider the use of online portfolios in our regular computer science undergraduate courses. The technical challenges we foresee include: the necessary support from the college's IT department; the use of Microsoft-based web authoring technologies vs. the use of Open Source / freeware counterparts; the need for adequate technical knowledge on the part of our faculty; the need for server-side hardware and software resources.

Categories and Subject Descriptors

D.2.6 [Programming Environments]: Integrated environments, Interactive Environments. I.7.2 [Document Preparation]: Hypertext/hypermedia, Markup languages, Multi/mixed media. K.3.2 [Computer and Information Science Education]: Computer science education, Curriculum, Self-assessment. K.8.1 [Application Packages]: Freeware/shareware.

General Terms

Documentation, Management, Performance, Human Factors.

1 INTRODUCTION

In the summer of 2004, we designed and ran a pre-college summer course in computing for high school students [1] with two major goals in mind:

1. To increase the visibility of our computer science program and improve recruitment.
2. To experiment with a new course format that is entirely lab-based, covers various areas of computing, and uses an online portfolio for student assessment.

The course material exposed students to web authoring and programming, databases, networking, computer organization, computer graphics, and Java programming. Lab activities were central to all the course modules, and were supplemented with field trips that brought real-world computing closer to student learning. The diversity of topics and class activities, as well as the compressed time format (six and a half hours of class time per day for six days) were clear challenges as to how we would provide students with feedback and what student assessment would best support learning in the course.

ISSN 1559-9388 (online version), ISSN 1559-9396 (CD-ROM version).

A natural solution to these challenges was to have students develop and use an online portfolio. The literature on portfolio-based assessment in higher education is vast [2]. The emergence of online portfolios in particular has recently been gaining larger acceptance [3], [4], [5], [6].

There are several factors that contribute to the wider adoption of online portfolios in higher education, and they characterize two current phenomena: on one hand, "online" is quite ubiquitous in today's instruction; on the other hand, emphasis on student-centered teaching is well served by portfolios, which have the advantage of supporting the process, rather than the product of student learning.

As a result of the steady rise of online learning, more and more colleges have adopted campus-wide course management systems. Consequently, course delivery has gradually shifted, completely or partly, to an online format. Secondly, the format of instructional materials and student work submissions is overwhelmingly digital. Finally, in computing education the knowledge areas of multimedia and web development, client-server technologies, databases, information systems, and ecommerce prepare students to understand and practice with web technologies.

In support of considering portfolios as a means of assessment in computing courses, we make the following observations. First, computing instruction in many courses consists of a large number of assignments, lab exercises, and projects that reflect the approach of learning by "doing" regularly and incrementally. Upper-level courses in software engineering, product management, capstone seminar, and others require students to manage and integrate multiple and diverse elements, not only programming exercises, of a significant project developed individually or in a team. Finally, active learning and student engagement with the subject matter by "doing" means little if instructors do not provide prompt, targeted, and continuous the instructor's obligation to incorporate formative feedback along the way make portfolios a very suitable assessment choice.

The adoption of student online portfolios for our summer pre-college course facilitated effective communication between course instructors and students, as well as uniform organization and management of student work:

- Students and instructors had a single location for recording evidence of work, including assignments, field trip and lab reports, and class notes.

All students were provided with a structure or template that standardized the organization of their work in the course and submissions for grading.

- Students could, in a convenient manner, take home and share with others the evidence of what they had learned.
- There was significant simplification and automation of the instructors' tasks of collecting, organizing, and storing all the student work produced in the class.

The college IT department assisted us with the configuration of a dedicated web server where all student portfolios resided. The software products we used in the course were all proprietary software and, with one exception, were Microsoft products. The course ran successfully and students learned and used effectively their online portfolios to showcase what they actually did in the course.

The rest of this paper is organized as follows. In the next section we describe our experience with configuring a course that uses student online portfolios for formative and summative assessment. We then explore the software choices that might replace proprietary software with Open Source or freeware software in all the areas we covered and needed in the course. We conclude with a summary of lessons learned and plans for future work.

ISSN 1559-9388 (online version), ISSN 1559-9396 (CD-ROM version).

2. ONLINE PORTFOLIO EXAMPLE

2.1 Design

The course content emphasized web-related topics so that the students could learn about real-world computing applications in a way that was relevant to today's job market. Also, to make the course as engaging as possible for the students, we gave it a very hands-on focus. As a result, we designed the course curriculum to include the following topics: Web Authoring and Programming, Databases, Networking, Java GUI Programming, Computer Organization, and Computer Graphics.

Incoming students did not typically have the necessary skills to know how to create a web site, so we structured the course to provide the necessary skills early on. Our first order of business was to have the students create a skeleton portfolio web site, from a template we designed for them. In order for the students to develop their web sites, we had to teach them some of the basics of web page creation and publication, so the topic of Web Authoring had to be first.

Subsequent topics built on this, and by the end of the course each student had completed an entire portfolio containing significant evidence of his or her own work.

2.2 Software

The students had, for the most part, very little experience with programming of any sort. Because of the brief and intense nature of the course, we needed to use software that used a graphical user interface to support a "what you see is what you get" (WYSIWYG) type of interface. This was easy for the students to use, and allowed the students to have a positive learning experience. In addition it provided strong feedback and a level of immediate gratification when they had accomplished a task.

Our college IT department already provides a number of software products, mainly from Microsoft, to all its faculty and students, and it was natural (and inexpensive) to choose some of these products for use during the course:

- Microsoft FrontPage, for web authoring
- Microsoft Access, for databases
- Microsoft Visual Studio, for C/C++ programming with OpenGL used during the Computer Graphics topic
- Borland JBuilder, for Java GUI programming
- Microsoft Internet Information Services (IIS), as the web server.

All of these products provide a visual learning environment, where students could see objects represented in a way they could understand and relate to.

In the particular case of web authoring, we wanted to focus on the content of each web page, and not on the details of the HTML tags required. While we did discuss the existence and purpose of HTML tags, we did not want the students to have to create web pages using such tags. Expecting the students to know and use HTML to construct their portfolios would inevitably lead to errors, frustration, and a shifting of focus away from page content to the irrelevant details of HTML tag-based document construction. Microsoft fits in very well with this requirement, as it provides a WYSIWYG interface to web page construction, while at the same time allowing access to the HTML tags, if needed.

ISSN 1559-9388 (online version), ISSN 1559-9396 (CD-ROM version).

2.3 Preparation

For the students to be able to create a useful online portfolio very quickly, we knew that they would have to start with a basic structure for their portfolio. To satisfy this need, we provided a template portfolio web site from which each student could tailor an individualized portfolio.

In addition, each activity that involved adding to, or modifying, the portfolio had to be designed and tested to ensure that things ran smoothly during the time-constrained course. For example, the students were asked to create a Microsoft Access database, and subsequently publish that database to the portfolio web site. Later, the students were asked to use FrontPage to construct web pages that allowed access to the data in that database.

2.4 Faculty Expertise

During the course of these preparations, one faculty member designed the basic structure of the portfolio as a FrontPage web, and then handed it to a second faculty member who gave it a representative look-and-feel, and added some other refinements, including the design of the databases topic, which involved uploading a database to the web site. This second faculty member also designed the activities involved in creating the student portfolio from this template, and provided detailed instructions for the students to follow.

Of the four faculty members teaching the course, one had a reasonably good knowledge of FrontPage, one some knowledge of FrontPage, and the other two faculty members tended to rely on the first two for web-related tasks.

It was important for at least one faculty member to have a good knowledge of web site construction with FrontPage. In addition, one faculty member was required to manage certain aspects of the web server, including setting up each student's portfolio area on the web site, and to maintain each student portfolio's autonomy.

2.5 Support

The support of our college's IT department was absolutely essential to the success of the course. We relied on them to:

- Create student accounts.
- Provide a server machine on which the portfolio web sites were hosted – we used a real web server, on a separate server machine.
- Provide support to address any hardware or software emergencies that could arise, during both the preparation and implementation phases of the course.

As is true for many colleges, our IT department is resource-constrained, and it could have been difficult for them to provide the necessary level of support for our needs. We were very fortunate to have a good relationship with them, and benefited from the extra support they were willing to give us.

We doubt strongly that we could have been successful in running this course without a very positive and active relationship with the IT department.

ISSN 1559-9388 (online version), ISSN 1559-9396 (CD-ROM version).

2.6 Results and Feedback

We found that our goals for having students develop and maintain online portfolios were well met during the course. The students created their portfolios and customized them with only a few minor problems (mostly related to not following instructions).

We were able to view each student's portfolio, and from it determine how well that student was progressing. We used the format of the portfolio as one contribution to each student's final grade in the course.

The feedback we received from the students was quite positive. We believe that the structure provided by the initial portfolio was a contributing factor to students' ability to organize their work successfully, and thereby complete the course despite considerable time constraints. While the students commented on the intensity of the course, we believe that without these structured portfolios the problems would have been much greater.

At the end of the course, parents attended a presentation by the students, who showed parts of their portfolios, and were clearly proud of what they had achieved. Students left the course with a flash memory stick containing a copy of their portfolio web site.

3. MICROSOFT VS. OPEN SOURCE SOFTWARE

One aspect of our experience that we decided to investigate further was the use of Open Source or freeware software products in place of Microsoft software products. Among the reasons for considering such a change are:

- **Cost:** While this was not an issue for our college, because it is already a Microsoft shop, it may well be an important issue for other colleges wishing to implement a web-based portfolio approach. There might be an impediment for students if student discounts of proprietary software are not available, especially in environments where students will not use college computers directly – for example, in distance learning situations.
- **Philosophy:** Some colleges and/or faculty members resist the use of Microsoft products for motivations other than cost. While this was not an immediate issue for us, we understand why it is important for many others.
- **Platform:** Typically, Microsoft products are only available on the Microsoft Windows platform. This would be a problem in those colleges who have standardized on systems such as Linux. In such situations, the choice of Open Source products would make more sense.
- **Exposure:** Open Source software is rapidly gaining in the market, with major vendors such as IBM supporting such activities. We believe that our Computer Science students need to be exposed to Open Source software at an early stage in their education, to be ready to compete in the job market.

3.1 Software Criteria

What criteria should we use to choose an appropriate set of Open Source software products? Based on the goals we had for our summer course, and our experience in designing and running that course, we consider that the following are the most important criteria:

- **Easy to learn:** Especially with beginning Computer Science students, there is a need to make things easy to learn, so that students can make good initial progress.

ISSN 1559-9388 (online version), ISSN 1559-9396 (CD-ROM version).

- **Easy to use**: Students can become easily frustrated if a software product is too complex for the task at hand.
- **Use direct manipulation of visually represented objects**: When objects can be visually represented and directly manipulated, students can understand the concepts more easily, and obtain a degree of immediate gratification upon completing an operation. This can encourage further investigation, and avoid frustration.

We feel that a GUI, WYSIWYG interface is essential for satisfying these criteria.

3.2 Open Source Choices

Open Source software is a very dynamic area. New Open Source products are being introduced, and existing Open Source products are evolving at dizzying speed. This complicates the task of evaluation of candidate products [5]. The best we can do is to take a snapshot in time of the most promising Open Source products that might meet our criteria and are relevant to the domains of web authoring, databases, Java, and web servers that we need for support of portfolios in computer courses.

This section provides information about some candidate Open Source products within these domains. It is a snapshot in time, and is not intended to be completely comprehensive.

3.2.1 Web Authoring

What should we choose as an Open Source replacement for Microsoft FrontPage? The advantages of FrontPage are:

- The ability to easily create web pages/sites without resorting to writing HTML tags directly. Users can construct a web page in a way similar to using Microsoft Word to create a document, using a WYSIWYG interface.
- The ability to easily publish web pages/sites to a web server, using either FTP or Front Page Extensions.
- The ability to create a web site with a consistent look-and-feel and navigation controls.

While there are many Open Source HTML editors available, we are looking for an Open Source WYSIWYG web page editor. This cuts down our choices quite a bit. Contenders include Nvu [7], Mozilla Composer [8], and Trellian WebPage [9].

Of these, Nvu [7] seems to hold the most promise. It is based on the Mozilla Composer technology, with some considerable usability improvements.

Trellian WebPage is freeware, but seems to be (about 3 years) old, and runs only on Microsoft Windows; it also requires an external program for FTP support, used for publishing the web pages to a web server.

One alternative to the use of a client-based WYSIWYG HTML editor is to use a server-based editor. That is, add web pages via a web server application. This approach, while having some precedence, will take much more work and investigation to see whether it is practical.

3.2.2 Database

What might we choose as an equivalent to Microsoft Access?

There are actually two components to this area:

- Database engine
- GUI.

ISSN 1559-9388 (online version), ISSN 1559-9396 (CD-ROM version).

Microsoft Access combines these two areas, but most database products do not.

3.2.2.1 Database Engine

Among the Open Source, or free, databases are: MySQL [10], PostgreSQL [11], Cloudscape (Apache Derby) [12], TinySQL [13], Hypersonic SQL [14], and Firebird [15].

MySQL is the leading contender: it is the most popular Open Source database, it runs on many platforms, and it is well documented, with a number of good books describing it. It is also relatively simple to administer.

PostgreSQL is a well-respected and popular database system that was developed on Unix systems. Recently, a Windows version has been made available.

Cloudscape's history is associated with Informix, and their acquisition by IBM. IBM has officially open-sourced Cloudscape and contributed it to the Apache Software Foundation, where it is now an 'incubator' project.

TinySQL, Hypersonic SQL, and Firebird are open source databases with smaller user bases.

There are, of course, many other database engines that could be used effectively.

3.2.2.2 Database GUI

Unlike Microsoft database products, most of the Open Source database engines do not include a GUI interface. Such interfaces are usually supplied separately, often by third parties. Java-based GUIs supporting the JDBC interface are particularly prevalent, given the ability of Java to provide platform-independent database and GUI interfaces.

Candidate database GUIs include: Eclipse [16] with the SQLExplorer Plug-in [17], iSQL-Viewer [18], SQuirreL SQL Client [19], QueryForm Database Tool [20], FreeQueryBuilder [21].

Note that all of the above candidates are written in Java, which indicates how strongly represented Java-based products of this type are in the Open Source area.

Probably any of these products would be satisfactory. None of them stands our significantly over the others in any obvious way. For students with a reasonable programming background, perhaps the use of Eclipse with SQL Explorer plug-in would be attractive. However, for environments where the students are not focused on programming (and sometimes even when they are!), the Eclipse IDE environment can be quite overwhelming, so one of the more self-contained products would probably be a better choice.

3.2.3 C/C++ Interactive Development Environment (IDE)

What do we substitute for Microsoft Visual Studio for the C/C++ programming and computer graphics areas?

Among the Open Source candidates are: Eclipse [16] with the CDT (C/C++ Development Tools) Plug-in [22], & GNU C++ [23], The V Portable C++ GUI Framework [24], and Qt [25].

Eclipse with the CDT (C/C++ Development Tools) Plug-in, & GNU C++ is probably the leading contender in this area, followed by the Open Source version of **Qt**.

3.2.4 Java IDE

For historical and legal reasons, Microsoft has no viable offering for Java. In our course, we used Borland's JBuilder because we are familiar with it, and because the college provides it on its machines. Replacement possibilities for a Java IDE include: Eclipse [16], NetBeans [26], Borland JBuilder [27], and BlueJ [28].

Eclipse is, again, perhaps the leading contender. As an Open Source project, it is acquiring a great deal of vendor support, and so is attractive for a number of reasons.

ISSN 1559-9388 (online version), ISSN 1559-9396 (CD-ROM version).

While not strictly an Open Source product, NetBeans is available free for non-commercial use.

Borland has a freely downloadable version of JBuilder that runs on a number of platforms, including Windows and Linux. JBuilder is also not strictly an Open Source product.

BlueJ is an interesting academic-inspired product that is focused on the effective teaching of Java. Many in academia find BlueJay an attractive product for this reason.

3.2.5 Web Server

With what should we replace Microsoft IIS? Because it is ubiquitous, proven, and well documented, the obvious Open Source replacement is: Apache HTTP Server [29].

The **Apache HTTP Server** is the most popular web server, is very well documented, and easily administered. It is the clear choice for a web server to replace Microsoft ISS.

3.2.5.1 Web Server support

One additional advantage that Microsoft FrontPage has is the existence of *FrontPage Server Extensions*. According to Microsoft, FrontPage Server Extensions "support features such as hit counters, data collection, e-mail processing, and database processing" [30]. The most important features we found were the automatic navigation updates and the easy configuration and modification of the overall navigation scheme of the web site.

When we move away from using FrontPage, we must find some way of providing equivalent functionality for those areas where we use these extensions. Basically, we need some kind of scripting support on the web server. The choices include: CGI [31], using Perl [32], or some other language**,** PHP [33], Java Server Pages and/or Java Servlets [34].

While CGI is built into the Apache Web Server, each of the other choices requires that we add some functionality to the web server.

For PHP, this is the Apache PHP plug-in. In the case of Java Server Pages/Java Servlets, Apache Tomcat is the most obvious choice for a Java Enterprise Container.

In addition, we will need to write the necessary code to substitute for the FrontPage Server Extensions, in particular to provide support for good navigation support.

4. CONCLUSION AND FUTURE WORK

In this paper we have described our direct experiences with requiring the use of an online portfolio in a summer pre-college course. We described our goals for such portfolios, and in particular, we focused on the resources and expertise necessary to implement portfolios in a successful manner. We learned that the practical implementation of such a portfolio provided significant benefits to both students and faculty members alike.

We have investigated a number of Open Source alternatives to the use of Microsoft software products, and, armed with the appropriate software setup, we can move towards requiring the use of online portfolios in our regular college courses.

We plan to work with these products to produce a viable approach to the use of portfolios in our regular computer science college courses.

We truly believe that the use of online portfolios can significantly enhance both the student and the faculty experience. The student can benefit from a more structured approach to providing evidence of the student's work, and the faculty member can leverage the use of portfolios to provide a more collaborative and interactive approach to learning.

ISSN 1559-9388 (online version), ISSN 1559-9396 (CD-ROM version).

4.1 Implementation in Computer Science Courses

We can expect students who are majoring or minoring in Computer Science to have, or to acquire, the necessary technical skills for them to use tools such as those described in earlier sections. In early courses, even Computer Science students will need some assistance, but the use of the above tools should not be confusing or terribly burdensome to them.

4.2 Portfolio Use in non-Computer Science Courses

It seems to us that the use of an online portfolio would also be very advantageous in non-Computer Science courses. Many – perhaps all – of our original goals for a portfolio translate easily into requirements for courses in other disciplines.

However, students who are not Computer Science majors cannot reasonably be expected to understand programming or the details of databases in order to create and populate a portfolio. For this population of students it is particularly important that we choose tools that they can understand and use, and that do not cause confusion or frustration. This is also true for the faculty members within those other disciplines, if we are to expect those faculty members to eagerly embrace such ideas.

5 References

[1] Sabin, M., Higgs, B., Riabov, V., Moreira, A. (2005), Designing and Running a Pre-College Computing Course. *Journal of Computing Sciences in Colleges, 20,* 5, 176-187.

[2] American Association for Higher Education and the University of Denver, Center for Teaching and Learning. *Portfolio Clearinghouse*. Accessed in June 2004 at http://ctl.du.edu/portfolioclearinghouse/.

[3] Estell, J.K. (2001). IPP: A Web-Based Interactive Programming Portfolio. *Proceedings of the 32nd SIGCSE Technical Symposium on Computer Science Education*, Charlotte, NC, 149-153.

[4] Abunawass, A., Lloyed, W., and Rudolph, E. (2004). COMPASS – A CS Program Assessment Project. *Proceedings of the 9th Annual SIGCSE Conference on Innovation and Technology in Computer Science Education*. Leeds, U.K., 127-131.

[5] Love, T. and Cooper, T. (2004). Designing Online Information Systems for Portfolio-Based Assessment: Design Criteria and Heuristics. *Journal of Information Technology Education*, vol. 3, 65-81.

[6] Yue, K.-B. and Ding, W. (2004). Design and Evaluation of An Undergraduate course on Web Application Development. *Proceedings of the 9th Annual SIGCSE Conference on Innovation and Technology in Computer Science Education*, Leeds, U.K.

[7] "Nvu: The Complete Web Authoring System for Windows, Mac, and Linux", http://www.nvu.com/

[8] "Mozilla Suite: The All-in-One Internet Application Suite", http://www.mozilla.org/products/mozilla1.x/

[9] Trellian WebPage, http://www.trellian.net/webpage/

[10] "MySQL: The World's Most Popular Open Source Database", http://dev.mysql.com/

[11] "PostgreSQL: The world's most advanced open source database", http://www.postgresql.org/

[12] "The Apache Derby Project", http://incubator.apache.org/derby/

[13] TinySQL, http://sourceforge.net/projects/tinysql/

[14] HSQLDB, http://hsqldb.org/

[15] Firebird, http://firebird.sourceforge.net/

[16] Eclipse, http://www.eclipse.org/

[17] SQL Explorer Plug-in for Eclipse, http://sourceforge.net/projects/eclipsesql

[18] ISQL-Viewer, http://www.isqlviewer.com/

[19] SquirreL SQL Client, http://sourceforge.net/projects/squirrelsql

[20] QueryForm Database Tool, http://sourceforge.net/projects/qform/

ISSN 1559-9388 (online version), ISSN 1559-9396 (CD-ROM version).

[21] FreeQueryBuilder, http://sourceforge.net/projects/querybuilder/
[22] CDT Plug-in for Eclipse, http://www.eclipse.org/cdt/
[23] GCC, http://gcc.gnu.org/
[24] V – A Freeware Portable C++ GUI Framework, http://programming.ccp14.ac.uk/objectcentral/main.htm
[25] "Trolltech – Qt Product Overview – single source C++ cross-platform application development for Windows, Linux, Mac", http://www.trolltech.com/products/qt/index.html
[26] NetBeans, http://www.netbeans.org/
[27] Borland JBuilder, http://www.borland.com/us/products/jbuilder/index.html
[28] "BlueJ – Teaching Java – Learning Java", http://www.bluej.org/
[29] The Apache HTTP Server Project, http://httpd.apache.org/
[30] About FrontPage Server Extensions from Microsoft. http://office.microsoft.com/enus/assistance/HP011032361033.aspx
[31] The Common Gateway Interface, http://hoohoo.ncsa.uiuc.edu/cgi/overview.html
[32] The Perl Directory, http://www.perl.org/
[33] PHP: Hypertext Preprocessor, http://www.php.net/
[34] Servlet 2.3 and JavaServer Pages 1.2 Specifications, http://www.jcp.org/aboutJava/communityprocess/first/jsr053/

---

* The paper was presented at the Association for Computing Machinery Special Interest Group for Information Technology Education Conference (SIGITE'05), October 20–22, 2005, Newark, New Jersey. It is published also in the Proceedings of the SIGITE'05 Conference, 2006, pp. 323-328. Published here with permission.

** **Dr. BRYAN HIGGS** is Assistant Professor of Computer Science at Rivier College. After earning his Ph.D. in Physics, Dr. Higgs entered the computing realm and worked in the computer industry for about 30 years. He spent 20 years at Digital Equipment Corporation, and seven years at Oracle Corporation. During that time, he obtained an M.S. degree in computer science from Rivier College and also became an adjunct faculty member at Rivier, teaching computer science courses in the evenings over a period of 15 years. In 2002, he joined the full-time computer science faculty at Rivier. Since then, he has developed and taught courses in Computer Organization, Java Programming, High Octane Java, C++ Programming, Perl Programming, Database Management Systems, Multimedia & Web Development, Client/Server Computing, and Computer Security. Dr. Higgs is a member of the Association for Computing Machinery. He is Chair of the Faculty Development Committee of the Faculty Senate.

*** **Dr. MIHAELA SABIN** is Associate Professor of Computer Science, Director of the Computer Science Programs, and Coordinator of the Mathematics and Computer Science Department. Computing and its many applications first became an important part of Dr. Sabin's life twenty-five years ago when she studied computer science at the Polytechnic University of Bucharest in her native country, Romania. After completing her master's degree, Mihaela researched the automation of writing programs, and developed tools that make computer-user interaction more friendly and natural. She pursued her doctorate in computer science jointly with a master's in college teaching at the University of New Hampshire. Along with more specialized research in the field of artificial intelligence and constraint satisfaction technology, Dr. Sabin became increasingly interested in being a teacher of computer science. Her current research is on the application of constraint satisfaction problem solving to network diagnosis, product configuration, and data storage management. She is also actively involved in advancing computer science and information technology education. Mihaela holds a strong belief that mathematics and sciences are an integral and essential part of today's education, research, and industry. Through her teaching and scholarship she is committed to promoting and improving the education of the young in mathematics, sciences, and computer science in particular.

ISSN 1559-9388 (online version), ISSN 1559-9396 (CD-ROM version).