

## A Student Designed, Web-based Learning Program for Circuit Analysis

Edward C. Shaffer and Frank J. Mabry  
 Department of Electrical Engineering and Computer Science  
 United States Military Academy  
 West Point, New York 10996

**Abstract.** *There are many topics in an undergraduate electrical engineering curriculum which are amenable to programmed learning. Traditionally, such instruction is achieved by means of a hard copy programmed text that the student works through, looking at solutions after completing an exercise and moving ahead or going back for remediation as necessary. Web based programmed learning has a particular advantage over traditional media because student actions can be captured electronically as the student works through the materials. This information can be used as valuable feedback for both instructor and student. The review and intelligent processing of the actions that the student takes to navigate through the programmed instruction can reveal important information regarding the nature and frequency of errors made. For example, an instructor can require all students to do web-based programmed instruction as a homework exercise; the subsequent aggregated feedback allows the instructor to pinpoint the most troublesome aspects of the material. Computer science students at the United States Military Academy (USMA) have designed a prototype web based application for performing electrical circuit analysis instruction as a senior design project. Details of the prototype and early experiences for applying it to an undergraduate engineering course are described here. Plans for future extensions of material development and management are summarized.*

### Background

This paper describes a pilot effort to develop a web based program that can be used to interactively assess student learning at United States Military Academy (USMA). The institution is fortunate to have an extensive, well established, networked computing environment designed to support academic requirements. Advances in web-based capabilities that reside on this academic network have enhanced opportunities for active learning. These observations are consistent with the NEA's recently announced quality measures for Internet-based distance learning [1].

Web-based learning exercises can be designed such that both students and faculty receive immediate, useful feedback about various aspects of an exercise. This feedback can be used for remediation, intervention, or to adjust process or

content. The focus of this particular effort was to have computer science students design an application to use sequences of learning (programmed instruction) to improve student learning in essential electrical engineering analysis and design tasks. One of the first problems that electrical engineering students taking an introductory course must solve is a multi-nodal dc circuit. A learning exercise was designed around an example nodal analysis circuit problem in order to illustrate the utility of the web-based approach.

An electrical engineering faculty member designed the example nodal analysis problem (content). However, the web-based "quiz engine" software and database were developed by the computer science students as project requirements. Conceptually, the students were to develop (1) database definitions that could be used to portray instructional quiz sequences and (2) the web based quiz-driver engine. The design was to be as "data-driven" as possible (see also "Question Objects: General Specifications" <http://ltsc.ieee.org/lib/questionobjects.htm>).

Close interaction was required between the faculty "client" and student project team to develop an interface and functionality sufficient to demonstrate proof of concept. The resulting software engine formed the basis for further development of this and similar projects, to include a "chemistry lab preparation system" and an "executive contract law annual review program".

The use of an extendible software engine overcomes limitations of other methods. Traditional computer based training is designed around a specific content area, which often involves extensive system overhead and cost. Once in place, it may be difficult for an instructor to adapt to student needs. At the other end of the spectrum, oversimplified web based systems are available, but these are usually very dependent on the specifics of the production computing environment, and management support and long term adoption or efficient reuse is unlikely [2], [3]. Although the specific problem to be solved here is a nodal analysis problem, the nature of the design allows the system to be readily modified for other types of electrical engineering problems (for example, mesh analysis, semiconductor biasing, Laplace or phasor circuit analysis, and power systems).

A number of software packages are available that perform circuit analysis. For example SPICE has commercial variants which include excellent interactive

graphical interfaces, allowing users to assemble component elements into a complete circuit and then to simulate the circuit performance, to include nodal voltage/bias point solutions. However, the mathematical solution steps are not apparent to the user. Mallard, a software engine developed by the University of Illinois ([www.cen.uiuc.edu/Mallard/](http://www.cen.uiuc.edu/Mallard/)), has been adapted by Bob Anderson at Iowa State University to support teaching of an introductory electric circuits course [5]. Although the use at ISU is similar to that intended at USMA, the implementation is different. It requires a dedicated Unix server running the proprietary Mallard software, whereas the prototype developed at USMA is designed to be open, expandable and modifiable using commonly available tools, and is designed to be platform independent.

### Nodal Analysis Example Problem

Electrical circuit analysis performed using the node voltage technique (nodal analysis) requires students to integrate basic knowledge (Ohm's law, Kirchoff's laws) into a more complex task. Electrical engineering instructors at USMA teaching the introductory course noted that they spent considerable time correcting errors made by students performing nodal analysis. In order to perform the analysis correctly, students must first generalize basic knowledge about the components and voltage and current interactions (Ohm's law). In addition to this declarative knowledge, students must understand the mechanics of the nodal analysis procedure (procedural knowledge [5]). Some of the more common errors students make include use of inconsistent voltage and current conventions, reversal of voltage and current signs, incorrect application of Kirchoff's current law, and incorrect application of Ohm's law.

The salient aspects of nodal analysis can be illustrated by using a three-node *ac* circuit. A student who successfully applies nodal analysis to a *dc* circuit of this complexity can usually generalize the procedure to *dc* circuits having more nodes, as well as to *ac* steady state circuit analysis. Note that *ac* steady state nodal analysis follows the same procedure but uses phasor analysis requiring complex numbers (conditional knowledge [5]). A sample three node circuit is shown in Fig. 1.

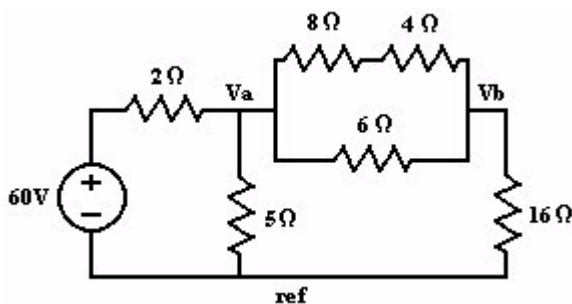


Fig. 1. Example Three Node Circuit Problem

Prior to performing the nodal analysis procedure on a circuit like this, the student must understand the following basic electrical concepts [6]:

- Voltage and current conventions
- Ohm's Law (*dc*):  $V_R = R \cdot I_R$
- Kirchoff's Current Law (KCL)
- Kirchoff's Voltage Law (KVL)

In order to perform nodal analysis, the following general procedure may be used [6]:

1. Define and label a reference node (usually where the most components are connected together).
2. Count and label the remaining independent nodes with a voltage designator with respect to the reference node.
3. Write the KCL equations at the labeled voltage nodes in a special form in terms of the node voltages.
4. Solve the simultaneous equations for the node voltages.
5. Use the solved node voltages to find any remaining unknown voltages or currents in the circuit by applying Ohm's law, KVL, or KCL

### Tutorial Considerations

It is apparent that a student can make any one of a number of errors in working through a nodal analysis problem to completion. A student who submits a meticulous written problem solution to the instructor in the form of homework or quiz can often receive sufficient feedback from the instructor to understand and correct these errors. However, a student often submits solutions that are missing steps, making remediation difficult and frustrating the student and instructor. In addition to improving the accuracy of the feedback, the timeliness of the feedback is also important. Accurate and immediate feedback to the student can be obtained via programmed instruction, usually in a self-paced form according to the ability of the student. However, traditional text based programmed instruction materials seldom provide feedback to the instructor or facilitator. Computer based instruction has the advantage that it not only provides immediate feedback to the student during a learning session, but also provides useful information to the instructor about the nature and frequency of errors.

The feedback available from computer based instruction can be particularly valuable when done on an aggregate basis. For example, an instructor assigns homework requiring completion of a problem that resides on a local web server. Each student must perform the assignment individually by logging in with a unique identifier. Information is gathered about each student's session on the web server. The choices students make are monitored, as well as the time between responses and the total solution time. The sequence of learning activities is structured in

such a way that the student is not frozen out from completing the assignment if he/she makes an error; after multiple errors of the same type, a correct solution to the step is provided and the exercise progresses to completion. The instructor can log onto the site anytime after the exercise has been assigned to gather information on an individual or group of students (in aggregate). The report provides information about the frequency and nature of the errors made, time latency of responses, completion percentages, etc. Analysis of this information provides an immediate assessment of learning, which can then be used to tailor subsequent classroom instruction.

Formulating the node voltage analysis tutorial in a form suitable for programming by the computer science project team required some negotiation between the client (instructor) and the student design team. The first two nodal analysis steps (identifying and labeling reference node and independent nodes) were to be given to the student. Completion of the remaining steps in the procedure included a series of multiple choice options. Incorrect choices were to provide an error message and require the student to try again. The number of errors allowed was to be limited; eventually the correct answer was to be given to the student and then he/she would be allowed to move on (limited repetition feedback paradigm).

## Software Design

This prototype application was developed using formalized procedures for the two-semester capstone design sequence in computer science at USMA. Objective of the software design process included:

1. Apply correct Software Engineering techniques to:
  - a. Develop a detailed design of a computer information system.
  - b. Build a computer information system.
  - c. Test the computer information system. Students develop and execute a viable testing plan to ensure system reliability.
  - d. Train the user of the system. Students develop and execute a realistic training plan to ensure that the ultimate user of the system understands how to use, modify and maintain the information system.
  - e. Manage the system development life cycle. Students learn to effectively plan, organize, lead and control the process of system development by effectively coordinating available resources.
2. Apply the concepts of corrective, perfective, adaptive, and preventive maintenance to the computer information system being built.

General requirements for the senior design project included:

1. General. Students work with a 'real world' client to design, build and test a computer information system.
2. Project Management. Critical to successful project delivery is effective project management. To accomplish this effort, student teams meet with the Project Director (Instructor), maintaining project logbooks, and maintaining a project schedule using project management software.
3. System. Design. The design phase focuses on the detailed design of the system. This phase culminates with a detailed design report and (if feasible) a working prototype of the system.
4. System Implementation. The implementation phase focuses on the development, testing, and delivery of the system. This phase includes: building and testing the system; training the user; final project delivery; and system maintenance.
5. Individual Performance Assessments. A variety of assessments are made with regard to individual work efforts within the student project groups. These methods include peer evaluations, client surveys/discussions, and instructor observations.

The original problem statement given to the computer science design team was broad in nature, beginning with the concept of providing web-based tutorial sessions for electrical engineering. The goal was to produce a program whereby EE students could login, select from a list of representative problems, present sufficient context for the student to solve the problem, guide the student through the solution steps, explain incorrect responses, and provide for instructor feedback and management. The student project team formulated the following minimum functional requirements of the objective "EE Cadet Problem Solving Tracker" system:

1. Input functions. The system must accept the following inputs:
  - a. Answers to EE Problems in multiple choice format
  - b. User Logon Information
  - c. Instructor Logon Information
  - d. Instructor Input of Problems and Solutions
2. Processing functions. The system must perform the following processes:
  - a. Compare entered student answers to the instructor's solution
  - b. Determine the amount of errors a student makes
  - c. Tally student responses in their respective categories

3. Output Functions: The system must generate the following outputs:

- Messages for correct responses
- Messages for incorrect responses
- Explanation for incorrect response
- Report how many students made each mistake
- Report how many students correctly answered a step in the problem

4. Storage Functions. The system will maintain the following data:

- Problems formulated by the instructor
- Responses given by each student
- Total correct and incorrect responses for each step
- Use information for each problem attempted

5. Control Functions. The system must enforce the following controls

- Verify problem solutions
- Verify user identity
- Verify administrator (instructor) identity

The computer science project team was required to develop several products in the formal software design methodology. Some of these included: a functional requirements contract; a software definition statement; risk identification and assessment; project schedule and software production milestones; data structure and data structure diagrams; sample nodal analysis problems; procedure descriptions; a scaleable “network student tutor” pseudocode; source code; problem files; tutorial guide; user’s manual; operations documentation; and future expansion of the project. Although details cannot be shown here, the functionality of the software will be illustrated.

The quiz engine software was implemented in ADA. Several ADA libraries were adapted and used to support a CGI (Common Gateway Interface) service that is platform independent [note: ADA can be compiled to JAVA byte code for operation across different computing platforms]. Student data input was kept simple in the first prototype by using multiple choice buttons presented via HTML forms. Text files were used to maintain the tutorial problem, state of the student, class identification numbers and passwords, and file location information. Screens for the users were produced by the ADA CGI program in standard HTML. Data files were saved on the web server’s file system in a secure area.

### Prototype Tutorial System

The resulting tutorial system designed by the computer science students has sufficient functionality to demonstrate “proof of concept”. Electrical engineering students and instructors use a browser to reach the local web server where

the system resides; upon access, a welcome screen appears with a login prompt. The software has been designed to allow the administrator or instructor to enter authorized user identifiers, which must be matched for a successful login. The student then selects a problem from a list of problems; in this case, it is the example nodal analysis problem. A screen then appears showing the problem (Fig. 2).

In using the prototype, the student works the problem on paper to find the correct KCL equations at nodes a and b, and then selects his/her response and submits it. If an incorrect response is given, an error message appears above the original screen and the student must retry. Once the correct set of equations is chosen, a new screen appears. It is assumed that the student will solve the correct simultaneous equations to find the nodal voltages (Fig. 3).

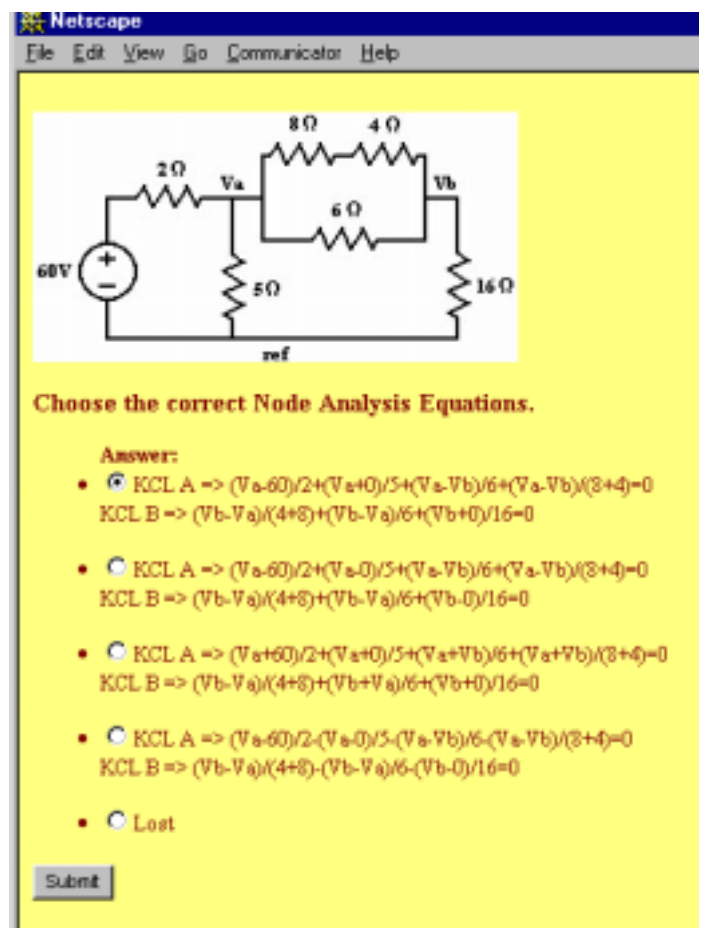


Fig. 2. Web Based Nodal Analysis Problem

Solving the simultaneous equations yields the values for nodal voltages; the next screen checks the student’s results; subsequent screens ask for the values of currents through particular resistors. The tutorial problem ends and records the data in a file. Other tutorial problems can then be solved

right

Which values are correct for the matrix?

Answer:

- W=0.9500, X=-.2500, Y=-.2500, Z=.3125
- W=0.9500, X=-.2500, Y=-.2500, Z=.3125
- W=0.9500, X=-.2500, Y=-.2500, Z=.3125
- W=0.9500, X=-.2500, Y=-.2500, Z=.3125
- Lost

Submit

Fig. 3. Intermediate Problem Step Page

or the student can try the complete sequence again without a new logon.

The instructor interface is somewhat different. In the prototype, problems are entered and edited off-line, through the browser. The performance of a particular student can be tracked either on line or by viewing the data file associated with that student. One very useful feature that has been implemented through the web interface is the ability to roll up results from multiple students. The instructor can determine how many students visited each screen, what their responses were, how many completed the exercise and the nature of errors (Fig. 4).

### Further Development

The “Network Student Tutor” software described here was originally designed to track student problem solving steps for electrical engineering, specifically nodal analysis. Since then

Student Data Histogram - Netscape

Problem Name: test problem 2

Number of students that answered this question: 10

A-5.....

B-7.....

C-4.....

D-3.....

Lost-3.....

A-2.....

B-3.....

C-5.....

D-4.....

Lost-2.....

A-7.....

B-3.....

C-2.....

D-1.....

Lost-3.....

A-0-

B-9.....

C-2.....

D-4.....

Lost-1.....

Fig. 4. Instructor Feedback Histogram

the software was modified into a “Chemistry Lab Preparation System”. The essential concept remained the same: a sequence of questions explore the declarative, procedural, and conditional knowledge associated with the steps of solving a complex problem using the right knowledge, in the right order. Students are expected to complete the exercise prior to performing an in-class laboratory experiment. By reviewing results of the exercise, the instructor can determine both the individual and aggregate level of understanding of the material and focus his efforts in clearing up muddy points. Additionally, the student who performs the exercise comes armed with sufficient background knowledge and predicted experimental outcomes before performing the lab.

There are many possibilities for further developing this prototype. A more sophisticated student interface, which accepts a variety of input data types, would enhance the experience. Embedded links to instructor developed or external materials could be used for immediate remediation

of errors (the present system already supports this). Web-based interfaces, which allow the instructor to edit the circuit graphics as well as problem statements, are also possible. The development of an XML based representation of the data is already underway. This work follows many of the functional inclusions from "Question Objects: General Specifications" [<http://ltsc.ieee.org/lib/questionobjects.htm>]. We plan to extend the definition to represent the type of cognitive knowledge being tested (declarative, procedural, or conditional). By inclusion of this information, instructor feedback can be improved by reporting students who make similar patterns of errors. For example, students who study at only a "surface" level material may answer declarative (factual) knowledge questions accurately, miss some procedural questions and completely fail on conditional knowledge questions. Diagnosing this type of learning problem would enable early instructor intervention and help avoid learner (and instructor) frustration.

Some additional work is planned to assess the effectiveness of the web based, interactive feedback design. An experiment is being designed to use the software engine within the software design course (CS407) at USMA. The experiment will use parallel classes with different instructors. A number of lessons will be supplemented using the web-based programmed instruction engine. Because the structure of class sessions at USMA are provide a controlled environment, there is high confidence in being able to determine the effect of classroom content delivery on the accuracy and timeliness of the web based response. Analysis of the collective statistics will allow assessment and redesign of various aspects of the web based learning software as well as how to better utilize it for engineering education.

## Conclusion

A prototype web-based tutorial system has been developed. Undergraduate computer science students designed the system, demonstrating that the division of labor between the content expert and the software system was not overly complex. The capability of using a web-based sequence of learning to provide feedback to students and instructors was successfully demonstrated. The first implementation of the system was used to solve an electrical engineering nodal analysis circuit problem. The software has been extended to another tutorial application in chemistry. Further improvements to the user interface promise to make the tutorial software a very useful tool for enhancing learning in a variety of academic disciplines.

## References

- [1] Lucivero, F. and Fleischauer, B. "NEA and Blackboard Inc. Study Finds 24 Measures of Quality in Internet-Based Distance Learning: Quality on the Line", 2000, URL: <http://www.ihep.com/PR17.html>
- [2] Spotts, T. H. "Discriminating factors in faculty use of instructional technology in higher education", *Educational Technology & Society*, 2(4) 1999.
- [3] O'Hagan, Chris (1999). "Embedding Ubiquitous Use of Educational Technology: is it possible, do we want it and, if so, how do we achieve it?" , *Educational Technology & Society*, 2(4) 1999.
- [4] Anderson. R.M. and Cheek, E.A., "Electric Circuits via the Internet: Sharing and Extending" Annual Meeting of the American Society for Engineering Education, 1990.
- [5] West, C.K., Farmer, J.A., and Wolff, P.M. *Instructional Design Implications for Cognitive Science*, 1991, pp. 15-18.
- [6] Cogdell, J.R. *Foundations of Electrical Engineering* 1996, pp.74-82.
- [7] Hallows, J. *Information Systems Project Management*, 1996, pp. 114-117.
- [8] Holcomb, R. and Johnson, M. "CS402 Capstone Design Project: Network Student Tutor," USMA, 1998.