Development and Implementation of a Distributed Virtual Laboratory for Continuous Manufacturing Education and Training

Gary E. Rafe, Kim LaScola Needy, Bopaya Bidanda
University of Pittsburgh

Ravi K. Ghai
HomeCareTraining.com

Therese A. Mylan
H.B. Maynard & Company

Introduction

A great deal of interest continues to be given to the potential of the global Internet to facilitate education and training in a wide range of disciplines. Following our earlier presentation of motivations for a *distributed virtual laboratory* (DVL),[1] we consider here the development of this distributed system and the context of its deployment in the training center of a Pittsburgh-based industrial engineering consulting firm. We begin by reviewing our definition of a *distributed virtual laboratory* in the context of continuous education and training for manufacturing. We next describe the mostly open-source system architecture of our DVL and summarize the implementation of the system's various components. We conclude this presentation by considering a case-study deployment of the DVL in a work measurement training course offered by H.B. Maynard and Company in Pittsburgh, Pennsylvania, in the spring of 2000. On-going and future development efforts are identified and described.

Background

In our earlier work, we defined the *virtual laboratory* as a "media-rich interactive environment of sufficient fidelity for conducting experimental activities associated commonly with some physical laboratory." The primary objective for this virtual laboratory is its ability to facilitate *experiential* learning support for Internet-delivered education and training programs, particularly in areas associated with the manufacturing discipline. Working closely with the Pittsburgh-based H.B. Maynard and Company, we identified topics within their work measurement continuing education course offerings that could be presented using our DVL.

An experimental Web-based training (WBT) player implemented as a Java *applet*[2] is used to deliver multi-media instructional content in the DVL, while a second applet, working in concert with software that implements the Virtual Reality Modeling Language (VRML)[3] provides dynamic, three-dimensional representations of our laboratory environment. We discuss these in greater detail later in the paper.

Distributed Virtual Laboratory Architecture

Our DVL employes a networked client-server architecture, as presented in Figure 1. This architecture relies on a highly-available *server* system capable of running a Structured Query Language (SQL) server, a Java 1.1 run-time environment (JRE), and a Hypertext Transport Protocol (HTTP) server that implements Java servlets. The physical server system hardware in our pilot implementation is a Sun Microsystems SPARCstation *IPX* running the Solaris 2.6 operating system. We chose the open-source MySQL[4] database system for several reasons; these include (1) its support for a large subset of the SQL92 language standard; (2) its extensibility through the use of user-defined functions on the server-side; (3) its ability to work transparently with proprietary Microsoft Windows-based applications using a freely-available Open Database Connectivity (ODBC) driver; and (4) the availablity of a compact and robust type 4 (*i.e.*, implemented completely in Java) JDBC driver that allows Java programs transparent access to the remote SQL server.[5] The popular open-source Apache Project's HTTP server[6] is used here primarily because of its integrated Java servlet engine, JServ.[7] Finally, we note that the Solaris operating system running on SPARC-based processors is the reference platform for Sun's Java run-time environment, which ensures the JRE's availability to implement our Java servlet.

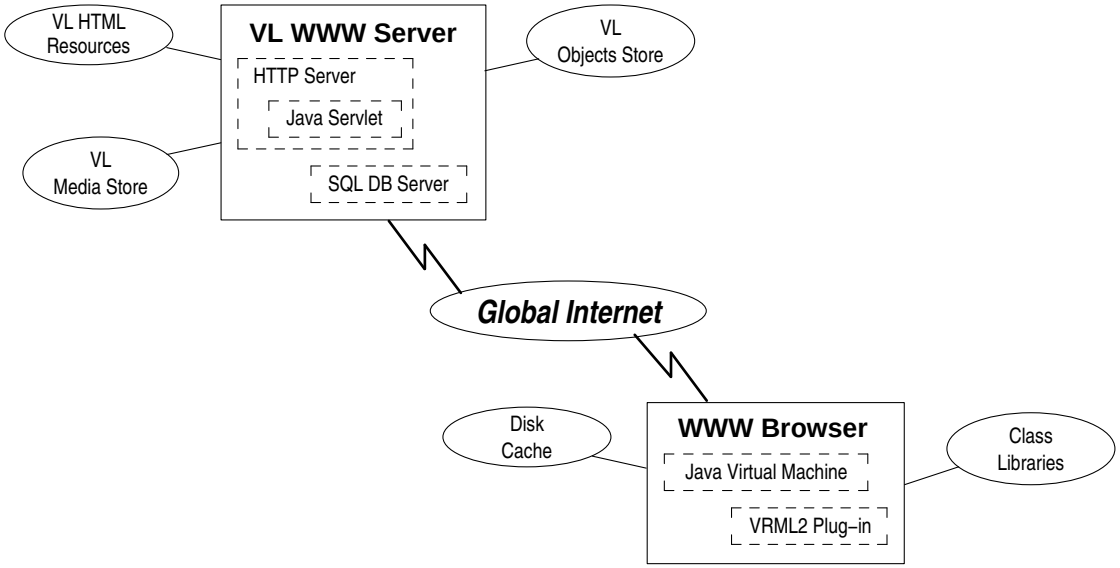

Figure 1 Distributed virtual laboratory architecture

A goal of this research is to realize an *architecture-independent* client environment. As indicated in Figure 1, the DVL's client requires a Java-enabled World-Wide Web (WWW) browser that supports VRML97-compliant plug-in software. A further requirement is support for the proposed VRML External Authoring Interface (EAI) extension by the VRML plug-in. In our present pilot implementation, the client is limited to Intel-based personal computers running the Microsoft Windows operating system due to the availability of required functionality in various hardware and software combinations. Current version of Netscape's *Communicator* WWW browser[8] and Platinum Technology's *Cosmo Player* [9] are used to implement our pilot client.

Textual and graphical instructional content are delivered to the distributed virtual laboratory clients by way of the MENTOR Web-based training (WBT) player, which is implemented as a Java applet. The MENTOR applet presents text, image, and other media objects arranged on *pages* that are organized in a *course*, *module*, and *lesson* hierarchy. A collection of *question* objects (*e.g.*, short text answer, multiple choice, and matching) can be used to establish varying levels of interactivity within a lesson. Responses to these questions are recorded in the database as they are completed, which permits the detailed tracking of student progress through their assigned lessons. Additionally, *evaluation* lessons containing collections of question item pages may be used to present graded evaluations. The results of these evaluations are also recorded in the remote database. Figure 2 illustrates the MENTOR WBT player applet running within Netscape's Communicator browser.

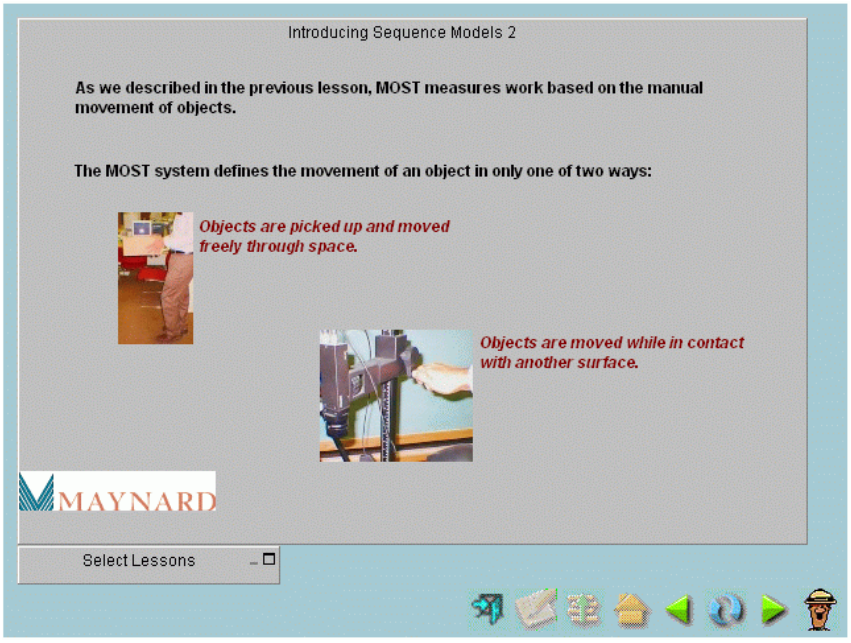

Figure 2 Java-based MENTOR WBT player

Instructional content for the WBT component of the DVL is created by way of a proprietary authoring program running on a Microsoft Windows-based personal computer. The content information created by this authoring program are stored in separate (but equivalent) SQL databases on the DVL's server. Following the creation of this instructional content, a UNIX shell script executed on the database server system is used to reconstruct the content information in a single database for use by the Java applets described here.

Virtual Laboratory Implementation

We next consider implementation issues relevant to the interactive virtual laboratory component of our system for continuous manufacturing education and training. We begin by offering a functional description of the virtual laboratory. The virtual laboratory is accessed within the courseware content from the WBT player by way of a *hyperlink* object embedded in an existing

lesson page. These hyperlink objects appear typically in the form of bit-mapped images, examples of which appear in Figure 2. Activating one such hyperlink object causes a new browser window to be opened, the address (URL) of which is a parameter of the hyperlink object. The destination parameter of this hyperlink object consists of two components. The first component describes the base Internet address (*i.e.*, URL) of the virtual laboratory servlet (*e.g.*, `http://vl_hostname/servlet_path/vl_servlet`); this servlet is described below. The second component of the hyperlink object is an optional *query string*, composed of a collection of *string-token* pairs that passes information about the subject's identity and course location to the virtual laboratory's servlet at run-time.

The HTML directives for the new virtual laboratory window are created dynamically at run-time by a Java *servlet*[10] process running on the remote virtual laboratory server, following authentication of the caller. The layout of this new window is dependent on the specific laboratory exercise, and is maintained in the virtual laboratory's SQL database. In general, the servlet presents the client browser with sufficient HTML code to describe an initially-empty embedded VRML scene, an associated controlling Java applet, and an embedded HTML `FORM` control to close the virtual laboratory window (Figure 3).

```
 1 <HTML>
 2 <HEAD>
 3 <TITLE>Virtual Industrial Engineering Laboratory</TITLE>
 4 </HEAD>
 5 <BODY>
 6 <CENTER>
 7 <EMBED SRC="runtimeVRML.wrl" BORDER="0" WIDTH="640" HEIGHT="320"
 8   MAYSCRIPT>
 9 </CENTER>
10 <CENTER>
11 <APPLET CODE="VielApplet.class" ARCHIVE="VielApplet.jar"
12   WIDTH="640" HEIGHT="140" MAYSCRIPT>
13   <PARAMETER NAME="viel_parameter" VALUE="viel_value">
14   <!-- additional Applet parameters -->
15 </APPLET>
16 </CENTER>
17 <CENTER>
18 <FORM METHOD="POST" NAME="closeForm">
19   <INPUT TYPE="button" NAME="closeButton" VALUE="Close Window"
20     ONCLICK="self.close();">
21 </FORM>
22 </CENTER>
23 </BODY>
24 </HTML>
```

Figure 3 Java servlet-generated HTML code

Prior to the delivery of the HTML code to the client browser, the servlet generates the initial VRML code referenced in line 7 of Figure 3. These codes are retrieved from the SQL database and stored in a temporary plain-text file. Temporary files created by the servlet process are removed periodically by a separate thread running within the servlet.

```
#VRML2.0 uft8
NavigationInfo { type { "NONE", "ANY" } }
WorldInfo {
  title "Virtual Industrial Engineering Laboratory: Lab 1"
  info "University of Pittsburgh 1999"
}
Collision {
  collide FALSE
  children [
    DEF VIEWPOINT0 Viewpoint {
      position 2.28 -12.41 14.05
      orientation -0.9147 0.1817 0.3609 -1.006
      description "Viewpoint 0"
    },
    DEF BACKGROUND0 Background {
      skyColor [ 0.6 0.6 0.6 ] groundColor [ 0.6 0.6 0.6 ]
    },
    DEF SPOTLIGHT0 DirectionalLight {
      direction 0.6 0.2 -0.2 intensity 1
      ambientIntensity 0.4 color 1.0 0.6 0
    },
    DEF ROOT Group { }
  ]
}
```

Figure 4 Java servlet-generated VRML code

The instructions illustrated in Figure 4 provide sufficient information to the client browser's VRML plug-in application for its initial setup. The accompanying Java applet presents (1) identification information (*e.g.*, exercise title); (2) one or more exercise scenes or configurations; (3) simulation action controls (*e.g.*, start, pause/stop, reset, *etc.*); (4) simulation output information (*e.g.*, timers); and (5) informational control(s) (*e.g.*, exercise-specific help display).

The applet makes SQL calls to the virtual laboratory database via JDBC[11] to instantiate VRML objects used in the specified laboratory exercise. We describe a schema for this database in the following section. As objects are added to the VRML scene graph hierarchy using the External Authoring Interface (EAI),[12] the identity and behavior parameters of objects participating in the simulation are recorded. Objects that have no behaviors associated with a particular exercise scene are rendered in that scene statically. Figure 5 depicts the distributed virtual laboratory's VRML scene viewer and Java applet running within the Netscape Communicator browser following its initialization.

Figure 5 Virtual laboratory VRML scene and Java applet panel

One of our research objectives considers the measurement of usage within the distributed virtual laboratory. To study this, we define two primary usage metrics within the DVL. These include trials per task (completed and uncompleted), and time per (completed) trial. *Tasks* are given by the configurations of an exercise. Exercises that have multiple configurations are composed of multiple tasks. A task *trial* is considered started after a configuration has been loaded into the DVL; a record is written to the database to mark this event. A task trial is considered completed when it reaches its *conclusion* (in this case, the simulation time at which no further events are pending); the trial start record described above is updated when the task trial reaches its conclusion. We also define the start of a new task trial when a configuration is *reset* following its earlier completion.

Virtual Laboratory Database Schema

The virtual laboratory exercises and their respective objects are maintained in an SQL relational database. A schema for the virtual laboratory is presented in Figure 6.

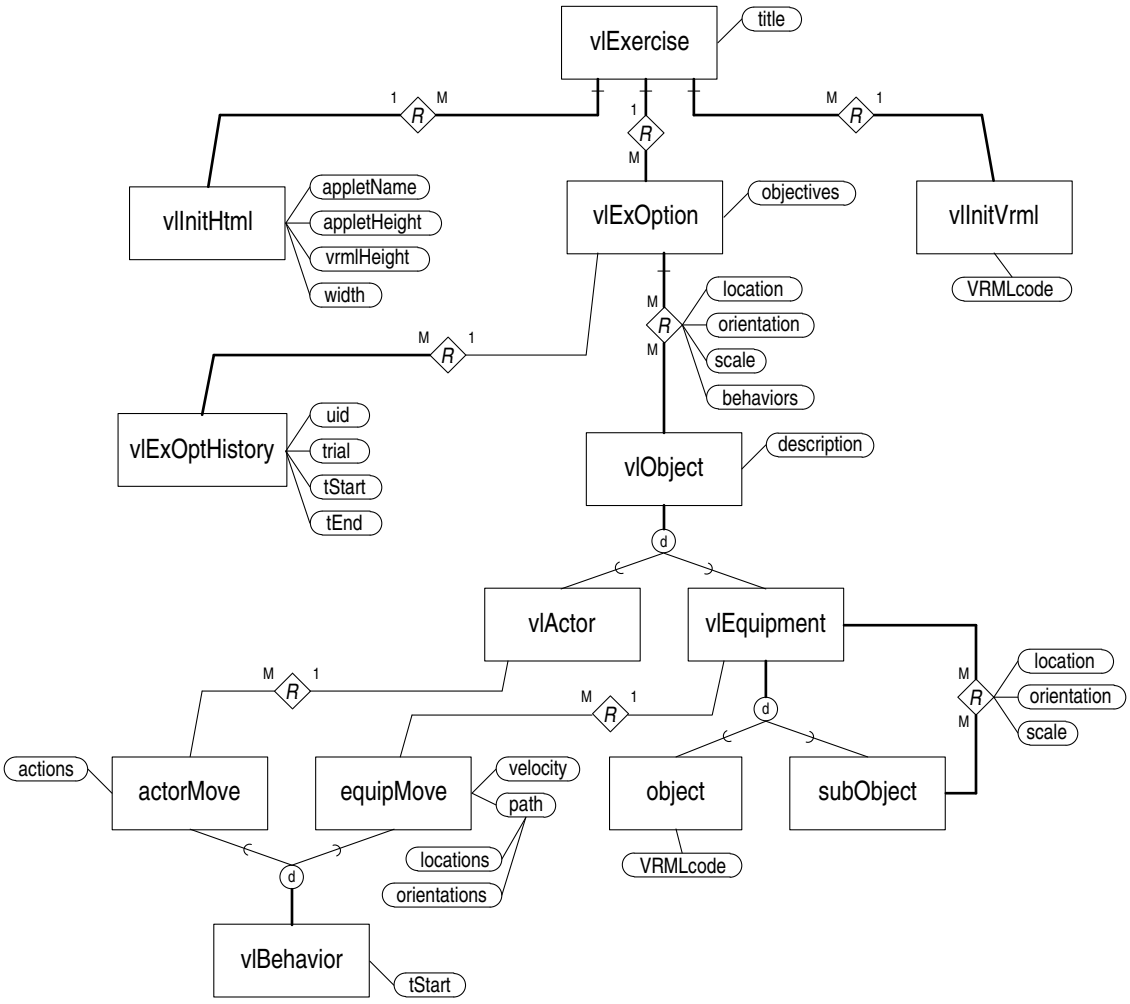

Figure 6 Distributed virtual laboratory database schema

The top-level entity in the DVL is the *vlExercise*. This entity is composed of at least one *vlExOption*, or configuration, and specifications for the initial HTML (*vlInitHtml*) and VRML (*vlInitVrml*) codes. These last specifications may be shared by multiple exercises. Each exercise option requires one or more *objective* attributes, which are displayed when the exercise option is loaded into the distributed virtual laboratory. The *vlExOptHistory* entity records start and end events of trials of exercise options for known DVL users (*uid*).

Each exercise configuration contains one or more *vlObject* entities. Instances of these entities are situated in the VRML scene with location, orientation, and scale information. An extensible set of *behaviors* associated with the object may be defined for individual objects in the exercise option. At present, only the generic *movement* behavior sub-class is defined; its implementation is discussed below.

Each *vlObject* entity is defined by one of multiple sub-class entities; we define at present the *vlActor* and *vlEquipment* sub-class entities. The *vlEquipment* entity is implemented here as a recursive set of *parent-child* relationships. Intermediate level *subObjects*, which act as sub-

assemblies of component-level objects, are implemented as simple VRML `Transform` grouping nodes. The lowest-level component objects are also given by VRML `Transform` nodes that specify the object's component geometries. The implementation of the *vlActor* entity and its associated behavior(s) are a subject of our current research effort. We are investigating how the proposed VRML Humanoid specification[13] can be applied in the context of our distributed virtual laboratory. Outcomes of this work will be reported when this paper is presented, and more fully in a subsequent paper.

We also note here that only deterministic movement sub-class behaviors in this pilot are defined at present. The *equipMove* behavior sub-class, which is associated with *vlEquipment* objects, has attributes *start time*, *velocity*, and *path*; the *path* attribute is composed of one or more segment end-positions, each recording location and orientation information. An extension to the distributed virtual laboratory will provide a higher-level task specification for the description of the movement of objects. This extension will accommodate the introduction of variability to event behaviors of primary mover objects (*e.g.*, actors), and allow event-driven actions, much in the fashion of discrete-event simulation environments.

When a scene is loaded into the DVL, root-level objects are found for that scene. After a root-level object is added to the VRML scene, any *child* objects are found and added to the parent object, recursively, until all child objects have been retrieved and added to the VRML scene. As objects are added to the VRML scene, we also build a list of VRML nodes that can be used to deconstruct the scene prior to the instantiation of a new exercise configuration.

We conclude our remarks regarding the distributed virtual laboratory's underlying architecture by noting that exercise configurations, their constituent objects, and object behaviors for the our pilot implementation are instantiated directly into the server's database using SQL scripts at present. We expect that the development of appropriate DVL *authoring* applications will be the subject of future research efforts.

Virtual Laboratory Case Study

We next consider a case-study deployment of the distributed virtual laboratory in the context of continuing education and training. The objective of this case-study deployment will be to assess the DVL's effectiveness in this setting. The industrial engineering management consulting firm H.B. Maynard and Company offers the week-long *Fundamentals of Work Measurement* course[14] at its Pittsburgh, Pennsylvania, training center approximately once per month throughout the year. The maximum enrollment for this course is 10. When sufficient demand exists, multiple sessions of the course may be offered, though not concurrently. Participants in this course come from various industry segments. Thus, we expect ample opportunity to study the DVL with a diverse subject population.

The instructional content instantiated in our virtual laboratory is presented by a facilitator in the first morning of the week-long course. Participating subjects will be invited to use the virtual laboratory's presentation during a scheduled break period just prior to a facilitator-led presentation of an introduction to the *Maynard Operation Sequencing Technique* (MOST) material.

The initial virtual laboratory exercise demonstrates aspects of a MOST *general move* sequence.[15] Users of this demonstration lesson are presented with a brief description of the virtual laboratory

space and the exercise by way of introductory comments and illustrations in the WBT player. Objects in the scene include (1) a small box, located initially on the floor; (2) a table; (3) an articulated *actor* standing several paces away, in the opposite direction. The exercise has one scenario (option). When the simulation is started, the articulated actor moves toward the box, bends and retrieves the box, rises with the it, then moves to the table and sets the box on the table. After the initial virtual laboratory window is closed, a brief review of the MOST general move sequence follows in the WBT player.

A second exercise adds a new configuration, in which the small box is located initially on a second table. When the second exercise is started, the initial scene is identical to the scene described above. A choice item control in the controlling applet offers the second scenario. When this configuration is selected, the initial scene is removed and replaced by the new scene. Following this brief exercise, MOST general move sequences are derived for the two scenarios and compared.

We conclude this discussion by noting that the development of appropriate qualitative and quantitative tools to assess the usability and effectiveness of our DVL in this case-study remain to be completed. We will be describing these assessment tools and the results of our assessment case-study in a future paper.

Summary

This paper described our recent efforts in the development and implementation of a *distributed virtual laboratory* for continuous manufacturing education and training applications using Internet-based information technologies. We also identify areas of on-going and future research and development efforts, including the implementation of an articulated actor and a higher-level task description for events occurring within a a particular exercise. Our presentation concluded with a description of a case-study deployment of the pilot DVL at the Pittsburgh training center of H.B. Maynard and Company in the spring of 2000. With the increasing interest in web-based education and training environments, we believe the distributed virtual laboratory described here will offer significant benefit to education and training content in the area of experiential learning, particularly to programs delivered via the global Internet.

Acknowledgements

We gratefully acknowledge the generous support of H.B. Maynard and Company, for access to the content of their *Fundamentals of Work Measurement* training course, and MCS, Inc., for the use of their MENTOR CBT (Microsoft Windows) WBT (Java) applications as the instructional content delivery component of our distributed virtual laboratory.

Bibliography

1. Rafe, G.E., K. LaScola Needy, & B. Bidanda. Motivations for a Distributed Virtual Laboratory for Continuous Manufacturing Education and Training. In *1999 ASEE Annual Conference & Exposition Proceedings.* Washington, DC: American Society for Engineering Education (1999).
2. Arnold, K. & J. Gosling. *The Java Programming Language.* Reading, MA: Addison-Wesley (1996).
3. Ames, A.L, D.R. Nadeau, & J.L. Moreland. *VRML 2.0 Sourcebook, 2nd Ed.* New York: Wiley (1997).
4. DuBois, P. *MySQL.* Indianapolis, IN: New Riders Publishing (1999).
5. Available at `http://www.worldserver.com/mm.mysql/`
6. Available at `http://www.apache.org/httpd.html`
7. Available at `http://java.apache.org/jserv/`
8. Available at `http://www.netscape.com/download/`
9. Available at `http://www.cosmosoftware.com/download/`
10. Hunter J., & W. Crawford. *Java Servlet Programming.* Sebastopol, CA: O'Reilly & Associates (1998).
11. Reese, G. *Database Programming with JDBC and Java.* Sebastopol, CA: O'Reilly & Associates (1997).
12. Marrin, C. *External Authoring Interface Reference* (21 January 1997). Available at `http://cosmosoftware.com/developer/moving-worlds/spec/ExternalInterface.html`
13. Roehl, B. *Specification for a Standard Humanoid* (3 August 1999). Available at `http://ece.uwaterloo.ca/~h-anim/spec1.1/`
14. *Fundamentals of Work Measurement Coursebook, 2nd Ed.* Pittsburgh, PA: H.B. Maynard & Co., Inc. (1999).
15. Zandin, K.B. *MOST Work Measurement Systems, 2nd Ed.* New York: Marcel Dekker (1990).

GARY E. RAFE

Gary Rafe is a Ph.D. candidate in the Industrial Engineering Department at the University of Pittsburgh. He received the B.S.I.E. degree from Alfred University, and the M.S. in Manufacturing Systems Engineering from the University of Pittsburgh. Previously, Mr. Rafe was on the faculty of the State University of New York's College of Technology at Alfred, teaching courses in computer and industrial control programming, computer-aided-design and manufacturing, and mechanical engineering technology, and managed the College's workstation laboratory network. His research interests include the application of information system technology in manufacturing enterprises, automating the product design-to-manufacturing process, CAD/CAM integration, and the use of telecommunication technology for training and education. Mr. Rafe is a student member of ASEE and IIE.

KIM LASCOLA NEEDY

Kim LaScola Needy is an Assistant Professor of Industrial Engineering at the University of Pittsburgh and a Wellington C. Carl Faculty Fellow. She received her B.S. and M.S. degrees in Industrial Engineering from the University of Pittsburgh, and her Ph.D. in Industrial Engineering from Wichita State University. She has obtained nine years of industrial experience at PPG Industries and The Boeing Company. Her research interests include Activity Based Costing, TQM, Engineering Management, and Integrated Resource Management. Dr. Needy is a member of ASEE, ASEM, APICS, IEEE, IIE, SME and SWE. She is a licensed P.E. in Kansas.

BOPAYA BIDANDA

Bopaya Bidanda is currently Ernst E. Roth Professor of Industrial Engineering at the University of Pittsburgh. He is a co-director of the University's Manufacturing Assistance Center and the Department's Automated Data Collection Laboratory. In addition to his doctoral degree from the Pennsylvania State University, he has industrial experience in manufacturing systems, tools, and precision manufacturing. Dr. Bidanda's research focus includes the areas of Computer Integrated Manufacturing Systems, Robotic Applications, Industrial Engineering, Automated Data Collection and Shared Manufacturing. He is a senior member of IIE and SME, and serves as a Director of the Pittsburgh Chapter of IIE.

RAVI K. GHAI

Ravi Ghai is Vice President of HomeCareTraining.com, a Division of Simione Central Holdings, Inc. HomeCareTraining.com develops interactive multimedia computer-based and web-based training and provides

content in generic topics (such as desktop applications, technology and soft skills) and home healthcare industry specific training in operational, financial, regulatory and clinical topics. In addition to the value-added content, it also provides tools and shells for the development, deployment and administration of interactive courseware. Mr. Ghai has spoken at many industry conferences, both domestically as well as internationally, in areas such as multimedia technologies, and electronic document management for manufacturing, workflow and interactive training. He holds the M.Engg and MBA degrees.

THERESE A. MYLAN

Therese Mylan is the Manager of H.B. Maynard & Company's Knowledge Center, in Pittsburgh, PA. As such, she manages a team that develops training and documentation materials and provides information resources throughout the company. Ms. Mylan received the company's President's Award in 1994 for her efforts in facilitating teamwork training at H.B. Maynard. She received the B.A. (English) and B.S. (Technical Writing) degrees from Carnegie Mellon University, and has seventeen years of experience in the technical writing and training fields.