Contents

ARTICLES

An Automated Curriculum Development Process for Navy Technical Training
Jeffrey A. Cantor 3

Developer Intent versus Instructor Delivery in Program Implementation
Cecilia L. Lopez, Howard J. Sullivan, and Sandra L. Weber 13

Instructional Development Models: Analysis at the Task and Subtask Levels
Robin Taylor and Philip J. Doughty 19

Teachers' Beliefs About Instructional Computing: Implications for Instructional Designers
Nancy Nelson Knupfer 29

Knowledge Mapping: A Multipurpose Task Analysis Tool
Timm J. Esque 39

DEPARTMENTS

ERIC Reports on ID, edited by Elena D. Beattie 51
An Automated Curriculum Development Process for Navy Technical Training

Jeffrey A. Cantor
Associate Professor
Lehman College
The City University of New York

Abstract. The need to contain the rising costs associated with the production of goods and services has posed challenges for instructional design professionals. This problem is particularly acute within the U.S. Navy's training commands. The outcome of this challenge has been Navy-sponsored research and development in computer-aided instructional design. This article describes some of these efforts.

There is increasing interest in automating curriculum development. The U.S. Navy is interested because (1) it teaches over 7000 different courses, (2) training materials and programs must be frequently updated, and (3) producing one hour of instruction involves 100 to 1000 person-hours at a cost of from $5,000 to $50,000. Experience has demonstrated that time savings of 50-75% can be achieved over conventional curriculum development through automation (Boucher & Goldman, 1986). The Navy undertook automating these efforts to:

- focus subject matter expert efforts more on subject matter rather than on computer operation,
- reduce many repetitive tasks and improve quality, and
- generate curriculum materials in camera-ready formats.

The Navy wanted a system that could:

- support both development and maintenance activities,
- operate with standard microcomputer hardware and word processing and communications software,
- be easy for technical subject matter experts to use, and
- comply with a new military instructional product development procurement standard identified as MILSTD 1375D.

A microcomputer-based program created for the Navy that automates curriculum development is described here.

The Process of Curriculum Development in the Navy

There are at least three forms of curriculum development in the Navy. One is factory training, whereby weapons system contractors develop related training as part of procurement. Another is generic training, whereby teams at Navy apprentice and advanced schools create non-system-specific courses, often with contractor support. A third is revision of existing courses by Navy instructors with contractor assistance.

The Navy's training and curriculum development processes are guided by the principles of instructional systems development (ISD) (Briggs, 1976; Cantor, 1987). These curriculum development efforts are guided by military procurement standards for Navy training programs supporting specific weapons systems.

Training Program Components

The automated instructional development program described in this article was initially developed to support the Navy's Strategic Weapons System Training Program (Cantor, 1985). To understand this automated instructional development application, it is necessary to first understand the particular instructional program components which the application supports. A brief overview of these program components is provided below, followed by the software development process.

Analysis and Design

The first phase of the Navy's automated development program involves ISD analysis and design activities. The software incorporates the structure, formats, and components to support the required training material development. The first step in the Strategic Weapon System Training Program is construction of personnel performance profile tables, which are comprehensive listings of the knowledge and skills required to operate and maintain a system or piece of equipment. (See Table 1.) These tables result from task analyses and are prepared using technical documentation and
### TABLE 1

**Background Knowledge and Skill Table Format (Knowledge)**

<table>
<thead>
<tr>
<th>Item No.</th>
<th>Knowledge/Skill</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.</td>
<td>KNOWLEDGE</td>
</tr>
<tr>
<td>1-1.</td>
<td>Define the terms, abbreviations, and symbols associated with basic electronics.</td>
</tr>
</tbody>
</table>
| 1-2.     | Describe the theory and construction features of semiconductors.  
  a. Diode  
  b. Silicon-controlled rectifier  
  c. Transistor  
  d. Field effect transistors  
  e. Unijunction transistor  
  f. Integrated circuits (chips) |
| 1-3.     | Describe operating characteristics of semiconductors. |
| 1-4.     | Describe the following types of transistor amplifiers and bias classifications:  
  a. Types  
    (1) Operational  
    (2) Audio  
    (3) Tuned  
  b. Bias classifications  
    (1) A  
    (2) B  
    (3) AB  
    (4) C |
| 1-5.     | Describe the operation of transistor amplifiers connected in one of the three basic configurations. Include relative advantages of each configuration.  
  a. Common emitter  
  b. Common base  
  c. Common collector |
| 1-6.     | Describe the theory and operation of basic types of oscillator circuits. |
| 1-7.     | Describe the operation and characteristics of particular oscillator. Include equivalent circuits and type of feedback circuit used in each of the following:  
  a. Armstrong  
  b. Hartley  
  c. Colpitts  
  d. Multivibrator  
    (1) Monostable  
    (2) Bistable  
    (3) Astable  
  e. Blocking |

Other pertinent data sources. Traditionally, performance profile tables were hand-prepared and where possible stored in computerized data bases. The tables are numbered corresponding to the system or equipment to which they are related.

Tables consist of two parts: knowledge items representing the theory, characteristics, functional operation and procedures involved in the operation and maintenance of the system; and skill items representing the abilities required to perform operation and maintenance based on acquired knowledge. Task items are written to encompass all levels (domains) of knowledge.

The next component, the training path system, assigns the occupational knowledge and skill items from the personnel profile to specific Navy classifications in a logical order of job performance and to a defined depth of required knowledge and skill level. The intent is to ensure that personnel assigned to specific training programs receive the necessary skills and knowledge commensurate with their assigned responsibilities. This training system component represents the design phase of the ISD model.

The training path consists of three subcomponents: training objective statements, training path charts, and training level assignments. The objectives define the level of training coverage for the knowledge and skill items to be learned. The training path chart presents a matrix of personnel performance items and objective levels required for each course, as well as the logical course sequence for specific personnel. The training level assignment component assigns the levels of training for specified knowledge and skill items to specific personnel, and also identifies the type of training (background, replacement, on-board, etc.). The software enforces development of a training level assignment chart for each profile within the proposed course. The process of building or modifying the training level assignment chart is greatly simplified and the possibility of error is reduced through use of rule-based algorithmic logic.

Training level assignment charts are an effective management tool within this training program. The training program components and material development process which the software supports are shown in Figure 1.
The software architecture consists of a series of input screen shells and prompts to assist the subject matter expert in entering task analysis data in response to questions asked by the computer program. This logic flow consists of a generic questionnaire input format which builds a mini database. Shells represent an empty template which is to be filled in by the operator. This information is then placed in a data base and used by the computer program to generate the output products. The computer programs supporting this automated tool are dBASE III Plus® and WordPerfect®.

The output of this first phase is the personnel profile tables, training analysis summary reports, and training path systems level assignments, including objectives and charts.

Materials/Curriculum Development

Curriculum development formats and materials constitute a detailed plan of organization for the formal presentation of information and the practice of skills. Curriculum materials are written for several types of Navy training, including background, replacement, advanced, maintenance, and conversion training. For group-paced, instructor-led training the curriculum materials usually consist of the topic learning objectives, instructor guide, trainee guide, testing materials, and instructional media materials.

Software Automation

Automated documents are developed through a series of text processing routines which are interactive with the relational database structure. All of the operations, such as specifying parameters and rules, sorting, and printing, are handled by macros. Macros are automated procedures which consist of stored command sequences appropriate for the applications being used for a particular function. These macros have been developed over time, based upon need as demonstrated through experience. For example, in the case of background and task/function personnel profile tables, a macro was designed to assist in the development of the table structure and in the selection of topic learning objectives and action verbs. For the training path system, the level/assignment generator provides for the file interaction necessary for objective arrangement into a topical outline permitting later sequencing in the curriculum development process. Once built, all training path data will be retained permanently in the training path system data base and integrated with the profiles and curriculum data bases for later use in building curriculum products. When operationalized by the report generator (a macro), the rule-based system produces tailored model statements to form the output report or table.

Each of the program files is designed to concentrate on a specific area of required information. However, in operation several files may actually operate interactively as a function is performed. This is possible through the use of the data base management software used in the design of this program. By way of example, the first of these program options to be used when constructing a profile will establish the profile number and ascertain whether the profile will contain classified material. Profile identification will continue with the establishment of the profile's name and occupational affiliation.

Unlike earlier attempts at this form of tool (Kearsley, 1986), a relational data base structure has been developed to alleviate data entry redundancy while
Figure 2. Personnel profile/training path system/level assignment development sequence—system flow.
progressing through input screens. Navy cost-consciousness dictates a need for expediency and elimination of excess labor time. This software tool was designed as an integrated system which allows the data produced by one phase of the ISD task to pass to another, whether forward or backward. This is most important, since a major effort in Navy curriculum development is maintenance of curriculum life cycle. File utility options allow all files created with the software to be stored and retrieved from a peripheral storage device connected to the hardware. Figures 2 and 3 graphically display the logic sequence for this automated process.

The software allows for the development of curricula materials in two phases. The first phase provides for the overall scheduling and planning of the curriculum. Through a series of text processing routines, a text sequence program is operationalized and data pertinent to the curriculum is automatically retrieved from the training path system file. Automated file selection of training level assignment and personnel profile tables is possible. This data is then processed through an objectives generator to develop a preliminary objective assignment chart which is displayed for the operator to review and modify as required.

Upon completion of the preliminary objective assignment chart, the system automatically generates the section titles, topic titles, and topic learning objectives. The operator is prompted to identify references, training materials and equipment that will be required to support each objective, and stores all data obtained for use in development of the preliminary master materials list.

All materials developed during the first phase are available for review and modification upon the completion of the development process. In practice, it is at this point that we as training contractors meet with the Navy for material reviews.

During the second phase of the development process, which provides for the preparation of all curriculum elements for the pilot course, the balance of the curricula is developed using the same data base created during the first phase of development. First phase revisions and corrections are made at this time. The second phase may be developed individually by part, section, topic, in any combination thereof, or in its entirety, as selected by the operator. The software system generator automatically generates a skeleton discussion point outline, using the topic learning objectives from each topic. This outline is available for review and modification by the operator.

Upon completion of the discussion point outline for each topic, the operator is prompted to identify references, graphics, instructional sheets, equipment, and training materials required to support each discussion point. Appropriate entries are automatically placed in the related instructor/trainee activity columns. As material requirements are added to the curricula, these items are placed on the appropriate topic page, under the heading of trainee/instructor preparation, and in the master material list as required.

The instructional sheets identified to support the curriculum are developed at the completion of each topic. All data previously entered and required during development is automatically

---

Figure 3. Training path system diagram.
Figure 4. Curriculum development sequence—system flow.

Legend:
- INPUT
- PRINTOUT

MINI-DATA BASE

(Courtesy Lockheed Systems)
placed on the instructional sheet. The operator is prompted for any remaining information required to complete the instructional sheet (e.g., study assignments, job steps). The software program runs on an IBM or compatible personal computer. It also has the ability to network with other computers when the hardware is configured with peripheral communications devices. The printer used is a graphics laserjet type, which permits fine-quality reproductions of technical subject matter.

Figure 4 displays the logic sequence for this automated process.

Management of Curriculum Development

The curriculum maintenance options available to the operator include automated statements, change preparations, review of materials, and approval of change materials. Based upon field trial experience, each option is now protected by a required password entry, and the level of access can be estimated for each user.

The automated statement provides the operator with the ability to search any group of materials (i.e., personnel profiles, training paths/training level assignments, curricula) and generates a list of all materials affected by the change(s). Personnel profiles and training paths/level assignment materials can be tracked for use in all other training materials.

Change preparation is accomplished by using the automated impact option which ensures that all occurrences of a noted deficiency are corrected. Alternatively, changes can be made by using the option which allows the operator to prepare a change recommendation for a single deficiency. The review of materials options provide the operator with the ability to review all current materials. Authorized users may, at their level of access, approve a pending change recommendation and incorporate this change into the master curriculum database. This option also generates an automated impact statement for the operator, so that all persons concerned will know where changes can be expected in the curriculum materials. Figure 5 displays the curriculum products relationships.

Discussion

The objective of the software project was to develop an automated, computer-based system for the (1) analysis and design of instructional materials for all types of Navy training; (2) development, preparation, and production of instructional materials; and (3) systematic support and management of the curriculum development process.

The many steps involved in the instructional design and development process require a set of tracking procedures for project management. This is not unique to the Navy, but applies to many organizations. Like most organizations, the Navy has budgets and time lines to follow in order to produce the training material. Project management requirements may vary from one training development activity to another, but most curriculum developers require basic management and tracking tools for their course development projects. This software package has the proven ability to support and assist in these tracking and management support activities. To take advantage of common course material(s) information, this training information must be added to and be accessible from a database. Access to paragraphs, topics, sections, and whole parts of already-developed courses would be quite useful to a developer working on a similar course. Time and money savings would be substantial.

In a recent application of this process to a curriculum redesign involving a Sonar Technician course, it was found that 80 hours of subject matter expert time was spent on the project in order to complete the necessary redesign. Since this course had been updated previously, a review of the records was undertaken. The last update took 240 hours of subject matter expert and instructional technologist time to complete. We have found that this automated capability gives us the advantage of producing a typical 40 instructional hour course in 45-66% less time. This results in increased profitability in a very competitive market, and supports Boucher et al.'s (1986) findings of substantial time and cost savings.

As graphic materials become electronic in format, they also must be organized in accessible data bases so that computer-stored graphic images can be available to other curriculum developers at the same training activity and to other training activities through the use of data base management computer programs. Many such software packages are able to support these needs.

The training materials development process involves a detailed audit trail through each of the design and development steps. For instance, there is a detailed methodology which specifies the links from the job tasks, to the objectives, to the presentation points in the lectures, to the testing materials. There is also the government requirement to ensure that changes in one part of the curriculum are accurately and automatically reflected in all other parts. Automation gives us the capability to be very accountable in our product development.

...this automated capability gives us the advantage of producing a typical 40 instructional hour course in 45-66% less time.
Figure 5. Curriculum products relationships.

Curricula Path Diagram

Curricula-Instructor Guide Diagram
The automated system is intended to better support Navy instructional planners and developers by reducing the time, effort, and expertise needed to produce curriculum materials.

The automated system is intended to better support Navy instructional planners and developers by reducing the time, effort, and expertise needed to produce curriculum materials, thus enabling us to become more competitive as a support contractor. The intent, therefore, is to optimize the process of instructional development and to standardize the products by working toward the following long-range objectives:

- Automate the formatting, production, and quality control standards of the numerous curriculum documents involved in training materials development.
- Provide a more customized software program to better assist in the various phases of curriculum development that require instructional expertise and curriculum development experience.
- Allow graphics editing and generation at workstations linked to laser printers.
- Provide a network environment with access to local and remote data bases so that existing instructional resources—both graphics and text—can be reissued with little or no revision.
- Integrate text and graphics in networked publishing environments.

This software continues to be tested and used in Navy applications. Its services continue to be demonstrated by quality control realized through automation, reduction in manhours of repetitive work, and enhancements of graphics.

Summary

The instructional systems software provides the subject matter expert with a powerful tool to develop a wide spectrum of training materials. Materials are developed with user-friendly menus which step the subject matter expert through the development processes. Materials developed are available for review, modification, and printing as menu options. Printed output is in the format specified by the operator.

The software was initially developed using existing commercial programs to allow Navy-formatted training materials to be developed and maintained in a fully relational data base and to allow for updates as training system development procedures and requirements are modified. The software is installed on an IBM PC compatible hard disk system. File utility options allow all files created with the software to be stored and retrieved from a peripheral storage device (i.e., external hard disk) connected to the hardware system.

The data bases created by the program include personnel profiles, training paths, training level assignments, curricula, and trainee guides. These data bases are linked to provide tracking throughout the training path. This linkage ensures that all training materials are developed and maintained as an integral unit, providing a means of identifying all materials affected by changes to other areas of the training program. These data bases are used for all development and maintenance functions of the program to ensure integrity and avoid unauthorized versions of the material.

References


Developer Intent versus Instructor Delivery in Program Implementation

Cecelia L. López and Howard J. Sullivan
Psychology in Education
Arizona State University
and
Sandra L. Weber
College of Engineering and Applied Sciences
Arizona State University

Abstract. Observation of the field testing of an instructional program for undergraduate engineering students revealed that the instructor did not collect or grade the student homework exercises as recommended in the instructor guide. Therefore, an experimental study was designed to investigate whether instructor delivery of the program as recommended would yield higher student achievement.

The study was conducted for six class periods with 19 students in the Fall semester and replicated with 14 students and a second instructor in the Spring. Homework exercises were collected, scored, and returned for the experimental group, as recommended in the instructor guide. The exercises were not collected for the control group, thus reflecting the procedure used by the instructor in the field test.

Posttest scores and grades were significantly higher for the experimental group. The test scores and grades by treatment were also consistent across the two semesters and two instructors. Self-report data revealed that experimental subjects spent significantly more time on the exercises.

Instructors often implement systematically designed instructional programs in a manner different from that intended by the developers....
Durzo and Florini (1984) stress that it is important to determine if the product was used as planned and to assess how what was done affected performance outcomes.

The present development and research effort involved a computer-based instructional program for university students majoring in construction engineering. The instructional program, entitled DOT Literacy (Weber & López, 1987), was originally developed by an instructional designer and a civil engineering subject matter expert (SME), using systematic instructional development procedures derived from the works of Sullivan and Higgins (1983) and Gagné and Briggs (1979). Its overall purpose was to teach students the use of the IBM disk operating system (DOS), version 3.1.

The instructor for the field test of the DOS program did not, in fact, follow certain key procedures specified in the instructor guide. The guide called for the instructor to assign homework exercises in the student book and to collect and grade them. Questioned after the field test, the instructor cited lack of time as his reason for not collecting and grading the exercises.

The designer suspected that the instructor's failure to follow the procedures for homework exercises as described in the instructor guide resulted in lower student achievement. Therefore, she designed and conducted an experimental research study after implementation of the program to determine whether using it as designed would yield higher achievement.

The study involved two treatments administered as part of the regular instruction in the university engineering course in which the DOS program was used. Under the experimental treatment, the homework exercises were collected and graded as directed in the instructor guide. Under the control condition, the instructor taught the program as in the field test, without collecting and grading the exercises.

The study was conducted across two semesters to yield additional data because the course enrollment was relatively small—15 to 20 students per semester. Both an experimental and a control group were employed during each semester. One instructor taught the program in the fall and another instructor in the spring in order to provide data on the replicability and generalizability of the findings.

The DOS Program

The DOS program was designed as part of a course for construction engineering majors entitled "Microcomputers for Constructors." The program consists of four modules, one for each of its four instructional objectives listed below in abridged form.

1. Use basic DOS commands (e.g., DISKCOPY, FORMAT, and SYS)
2. Use advanced DOS commands (e.g., BACKUP, RESTORE, and CHKDSK)
3. Create and manipulate fixed disk subdirectories (e.g., MD, PATH, TREE)
4. Generate batch files, using batch file commands (e.g., ECHO, %n, GOTO) and DOS commands.

The program requires approximately seven hours of class time to complete. Homework assignments take several additional hours.

The format for each of the four modules follows a similar sequence. The instructor introduces the module and previews its content for the students. Basic information is presented both by the instructor and in the student book. The instruction for each module in the student book incorporates 15–20 practice questions with feedback, followed by a review of the purpose and syntax for each command. A homework exercise on the computer is included at the end of each module. The instructor guide directs the instructor to assign each homework exercise at its appropriate point. The program also includes a 40-item constructed-response posttest consisting of 10 items covering the instructional objective for each module.

The DOS program was field tested under regular classroom conditions in the spring of 1987 in the engineering course for which it was developed. The mean percentage score of only 5% on the pretest, a 40-item measure in a form equivalent to the posttest, revealed a definite need for the program. The mean posttest score for the field test was 61%. Observation of the field test by the instructional designer revealed that the field test instructor, who was also the regular course instructor, assigned the homework exercise for each module but did not collect and grade the exercises as described in the instructor guide.

Both the SME and the course instructor were pleased with the field test results and decided to incorporate the DOS program into their course. The instructional designer, however, hoped to improve student performance and identified use by the instructor of the specified homework procedures as a possible means of doing so. She therefore designed an experimental study to investigate whether the instructor's use of these procedures would, in fact, yield higher student achievement. The DOS program was incorporated into the regular "Microcomputers for Constructors" course beginning in the fall of 1987, and this course provided a natural setting for the designer to conduct the investigation.

...it is important to determine if the product was used as planned and to assess how what was done affected performance outcomes.
Method

Participants

Subjects were 33 undergraduate Construction Engineering majors enrolled in the “Microcomputers for Constructors” course in the School of Engineering at Arizona State University during the fall of 1987 (N = 19) and spring of 1988 (N = 14) semesters. Both the regular course instructor and the SME are Associate Professors and licensed Professional Civil Engineers.

Materials

The instructional program was the previously described DOS literacy program on the disk operating system for the IBM microcomputer. The program was revised on the basis of field test observations and data prior to its final printing for use beginning in the fall of 1987.

Both the experimental treatment, in which the homework exercises were collected and scored, and the control condition, in which the exercises were not collected and scored, were employed each semester on a between-subjects basis. For each semester, subjects were assigned randomly to the two treatment groups. The regular course instructor taught the program during the fall semester. The SME substituted for the regular instructor for the DOS portion of the course during the spring semester. Classes met twice a week for an hour and 20 minutes, and instruction for the program took five class sessions.

The classes were informed that, as part of an experiment, the instructor would collect and grade the computer exercises of certain students, who were then identified by name, but not of the other students. The students whose work was to be collected and graded turned in printouts of their exercises at the beginning of the class period following each assignment. The experimenter scored the exercises for the instructor and wrote brief comments and grades on them. Exercises were not collected from the control students. The experimental subjects’ exercises were returned to them at the next class session after they were submitted. The answers for the exercises were distributed to all students, both experimental and control, after the experimental group’s work was returned. Answers to the exercises were incorporated into the student test in the field test, but were removed from the revised version of the text on the recommendation of the instructor and several students.

The posttest and an attitude questionnaire were administered to all subjects during the sixth class period.

Criterion Measures

The primary criterion measure was the 40-item constructed-response posttest which covered the same instructional objectives as the practice questions and the computer exercises. Posttest inter-item reliability calculated with Kuder-Richardson formula 20 was .88.

The grades which instructors assigned to students for the program provided the basis for a second criterion measure. The instructor for each semester assigned the grades from test scores without knowledge of the subjects’ names and, consequently, of their particular treatment group. Grades supplemented the posttest as a criterion measure by providing a correlated, but more global, measure of the importance of any observed differences in test performance.

The 10-item attitude questionnaire assessed students’ attitudes and perceptions related to the assignments. The items dealt with matters such as students’ willingness to do computer exercises on other topics, their preferences related to having the exercises graded, the time they spent on the exercises, and how important they thought the exercises were.

Design and Data Analysis

The experimental design was a randomized posttest-only control group design (Campbell & Stanley, 1963, Design 6). Posttest achievement was analyzed by t test, and questionnaire items were analyzed individually by chi-square.

Posttest scores were very similar within treatments across the two semesters and instructors, and t tests of the within-treatment differences across semesters revealed that they were not significant for either treatment. Data were therefore pooled across semesters for the between-treatment analyses.

Results

Posttest Performance

Table 1 shows that the mean posttest scores were 25.83 (65%) for the experimental group and 20.88 (52%) for the control group. This difference was statistically significant, t (131) = 2.08, p < .05.

The table also shows that the greatest between-treatment difference within a module occurred in Module 4 (generating batch files), which requires the most complex student performance. Mean scores for this module were 5.63 (56%) for the experimental group and 3.35 (34%) for control, also a statis-
TABLE 1
Means and Standard Deviations of Test Scores by Treatment and Time of Experiment

<table>
<thead>
<tr>
<th></th>
<th>EXPERIMENTAL</th>
<th></th>
<th>CONTROL</th>
<th></th>
<th>Experimental-Control Difference (E-C)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Posttest</td>
<td>Fall 1987</td>
<td>Spring 1988</td>
<td>Total</td>
<td>Fall 1987</td>
</tr>
<tr>
<td></td>
<td>Scores</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Module 1</td>
<td>M</td>
<td>7.11</td>
<td>7.14</td>
<td>7.13</td>
<td>6.50</td>
</tr>
<tr>
<td></td>
<td>SD</td>
<td>1.27</td>
<td>2.41</td>
<td>1.78</td>
<td>1.65</td>
</tr>
<tr>
<td>Module 2</td>
<td>M</td>
<td>6.33</td>
<td>6.43</td>
<td>6.38</td>
<td>5.40</td>
</tr>
<tr>
<td></td>
<td>SD</td>
<td>1.50</td>
<td>1.61</td>
<td>1.59</td>
<td>1.96</td>
</tr>
<tr>
<td>Module 3</td>
<td>M</td>
<td>6.89</td>
<td>6.43</td>
<td>6.69</td>
<td>5.40</td>
</tr>
<tr>
<td></td>
<td>SD</td>
<td>2.76</td>
<td>2.88</td>
<td>2.73</td>
<td>1.78</td>
</tr>
<tr>
<td>Module 4</td>
<td>M</td>
<td>5.11</td>
<td>6.29</td>
<td>5.63</td>
<td>3.10</td>
</tr>
<tr>
<td></td>
<td>SD</td>
<td>1.76</td>
<td>2.98</td>
<td>2.36</td>
<td>2.47</td>
</tr>
<tr>
<td>Total</td>
<td>M</td>
<td>25.44</td>
<td>26.29</td>
<td>25.83</td>
<td>20.40</td>
</tr>
<tr>
<td></td>
<td>SD</td>
<td>4.64</td>
<td>8.08</td>
<td>6.15</td>
<td>6.47</td>
</tr>
</tbody>
</table>

cally significant difference, $t(1,31) = 2.84, p < .01$. The experimental group also scored higher than the control group on each of the other three modules, but these scores did not differ significantly between treatments.

Grades

The grade distribution for the program summed across the two semesters is as follows:

<table>
<thead>
<tr>
<th>Experimental</th>
<th>Control</th>
</tr>
</thead>
<tbody>
<tr>
<td>A: 4</td>
<td>A: 2</td>
</tr>
<tr>
<td>B: 8</td>
<td>B: 4</td>
</tr>
<tr>
<td>C: 3</td>
<td>C: 6</td>
</tr>
<tr>
<td>D: 1</td>
<td>D: 4</td>
</tr>
<tr>
<td>E: 0</td>
<td>E: 1</td>
</tr>
</tbody>
</table>

These grades convert to an overall GPA of 2.94 for the experimental group and 2.12 for the control group. A $t$ test revealed that the difference was statistically significant, $t(1,31) = 2.37, p < .05$. Grades within treatments were similar across the two semesters: 3.0 for experimental subjects and 2.10 for control subjects in the fall, and 2.86 for experimental subjects and 2.14 for control subjects in the spring.

Student Attitudes

Student responses to the 10-item questionnaire revealed significant differences favoring the experimental group over control subjects on two items, time on task and perceived level of task difficulty. Sixty-three percent (10 of 16) of the experimental subjects and 24% (4 of 17) of control subjects reported spending from three to five hours or more on the computer exercises for each module, $\chi^2(1,31) = 5.12, p < .05$. The remaining subjects reported spending less time on the exercises. Sixty-five percent (13 of 20) of the experimental subjects and 41% (7 of 17) of control perceived the computer assignments as difficult, $\chi^2(1,31) = 5.54, p < .02$. The attitudes of the experimental group were more positive, but not significantly so, on six of the remaining eight questionnaire items.

Discussion

The overall results of the study reveal that delivery of instruction as described in the instructor's guidelines yielded significantly higher student posttest scores and grades in the experimental treatment. The critical instructor behavior in producing this effect was the collecting and grading of assigned homework. Student reports indicated that the same instructor behavior also resulted in greater student time spent on the homework task.

The most likely dynamics underlying the performance differences relate to the fact that experimental subjects knew that their work would be collected and graded, whereas control subjects knew that theirs would not. Consequently, experimental subjects spent more time on the exercises and thereby had more practice on the instructional objectives covered in them and assessed on the posttest. Both practice and time on task have been found to improve learner performance (Craik & Lockhart, 1972; Popham, 1969; Salisbury, Richards, & Klein, 1985). In addition, the experimental group also received specific feedback on their performance on the exercises and not just the answers subsequently given to both groups.

The strong effect for Module 4 is most likely a function of the nature of the content of that module. The skills learned in Modules 1–3 were prerequisite skills for Module 4, which required much more complex behavior than the preceding modules. In Module 4, students had to use the DOS commands learned independently in Modules 1–3 and integrate them in the proper order with the new commands from Module 4 to construct a batch file. This is a complex task that requires application and synthesizing of new information with prior knowledge.

The increased practice by the experimental group could be expected to have its greatest effect on complex learning tasks such as the one in Mod-
ule 4. Sullivan and Higgins (1983) stress the importance of more frequent and individual practice for conceptually difficult learning tasks. Research by Craik and Tulving (1975) and Bretzing and Kulhavy (1979) indicates that practice in elaborating and organizing previously learned information yields better performance in recalling the information.

Subjects whose exercises were collected and graded in the experiment scored 13 percentage points higher (65% as opposed to 52%) than those whose exercises were not collected and graded. However, students in the field test the preceding year had averaged 61%, only four percentage points lower than the experimental group and nine points higher than the control group, even though their instruction was like that of the control group. This apparent anomaly led the investigators to question the course instructors and examine student records for possible explanations.

The course instructor reported that the students in the field test class were highly motivated, much more so than the classes in the experiment. He explained that the semester of the field test was the first time the class had been offered, and only subjects who wanted to take it enrolled because it was strictly an elective course at that time. Students who took the course during the year of the experiment were less self-selected because it was known that the course was soon to be made a requirement, which it now is. Examination of students’ records revealed that the cumulative undergraduate grade-point average for the field test class was 2.72, as contrasted with 2.53 and 2.57 for the two classes in the experiment.

The present results highlight an important difference of opinion between instructional designers and course instructors that is not unique to this study. Being well versed in the precepts of competency-based instruction, the instructional designer was not satisfied with posttest scores in the 60% range, just as many developers would not be. She had worked very hard to develop the program and, to a large degree, felt personally responsible for student mastery of its content. She wanted higher student achievement and felt that one way to achieve it was to have present and future course instructors collect and grade the homework exercises. She also recom-

mended extending the length of the program to provide more student practice and instruction, a recommendation that was also made by several subjects in the field test and experiment.

The regular course instructor had a different opinion and was refreshingly candid about it. He was satisfied with the level of performance from the field test. During debriefing by the developer, he explained that he considered the learning task to be an important but difficult one, and that the task was very difficult. “The easiest way to get from 60% to 80% or higher,” he noted, “is to change the test. Make it easier.”

The instructor reported that he was too busy to grade the exercises as specified in the instructor guide, but that he would probably do so if he had more time or a grader for the course. He also was unwilling to allocate more time to the program because of the amount of content to be covered in the rest of the course.

Who’s right? Certainly both sides have their advocates. The developer’s opinion is consistent with that of instructional designers generally and with current models of competency-based instruction and mastery learning. Instructional developers are trained to attempt to produce consistently high learner performance, and they generally place greater responsibility on themselves, and perhaps on the instructor, for student learning.

The instructor’s opinion, on the other hand, reflects an attitude of many university faculty members and is quite common in disciplines such as mathematics and the hard sciences, including engineering. Many faculty do not ex-

Who’s responsible for student learning?
...it is important for instructional designers to recognize the practices and perspectives of the types of instructors who may use the programs that the designers produce.

References


Instructional Development Models: Analysis at the Task and Subtask Levels

Robin Taylor
Department of Education
University of Maryland Baltimore County

Philip L. Doughty
Area of Instructional Design, Development & Evaluation
Syracuse University

Abstract. A simple yet detailed process for analyzing and comparing instructional development models at the task and subtask levels is described, justified, and illustrated. The analysis process ensures that all major instructional development tasks and their subtasks are identified, assigns weights to the subtasks, and facilitates the comparison of different prescriptive models task by task, according to both comprehensiveness and operational level.

In a review of instructional development models, Gustafson (1981) noted that models can "serve a variety of purposes, including theory building and testing, description, prediction, and explanation" (p. 4). He also considered three principal uses of models by instructional developers: as communication devices, planning guides, and prescriptive algorithms. In this article, the usefulness of instructional development models as planning guides or management tools is considered. Viewed from this perspective, according to Gustafson, models should account for all of the major tasks to be performed" (p. 4).

In this article, a new process for analyzing instructional development models is explained and then illustrated for five selected models.

Instructional Development Defined

Before examining how models for the development of instruction can be analyzed, compared, and contrasted, it may be useful to outline the fundamental activities involved in instructional development. The complex set of processes known as instructional development or instructional systems development is illustrated by Carey and Briggs' (1977) simplified "design" model, shown in Figure 1.

As used in this article, instructional development is defined according to Silber (1977):

...a systematic approach to the design, production, evaluation, and utilization of complete systems of instruction, including all appropriate components and a management pattern for using them.

Instructional development is larger than instructional product development, which

---

Figure 1. Simplified instructional development model (from Carey & Briggs, 1977, p. 284). The project flow is from the top to the bottom of the diagram.

<table>
<thead>
<tr>
<th>ANALYSIS</th>
</tr>
</thead>
<tbody>
<tr>
<td>DESIGN</td>
</tr>
<tr>
<td>DEVELOPMENT</td>
</tr>
<tr>
<td>FIELD TRIAL</td>
</tr>
<tr>
<td>INSTALLATION</td>
</tr>
<tr>
<td>DIFFUSION</td>
</tr>
</tbody>
</table>
Instructional development encompasses all of the activities that move a project from its conception to the implementation of an appropriate instructional system in a desired context.

is concerned with only isolated products, and is larger than instructional design, which is only one phase of instructional development. (p. 172)

Instructional development encompasses all of the activities that move a project from its conception to the implementation of an appropriate instructional system in a desired context.

Instructional development models presented in the literature have a variety of origins and purposes. For this article, only the development of elementary- and secondary school instructional materials is considered. Therefore, only tasks specifically directed toward creating instruction for those contexts have been examined. The phases of Analysis, Design, Development, and Field Trial in the Carey and Briggs (1977) model are relevant to this article. Responsibility for the Diffusion and Installation phases usually falls to school systems or to sales and marketing groups.

Previous Analysis Methods: Model and Task Levels

Two well-known reviews of instructional development models (Andrews & Goodson, 1980; Stamas, 1973) present “macro-analysis” comparisons. In both reviews, instructional development tasks are described in very broad terms, and each model is analyzed with respect to all tasks. In neither review is the amount of explanatory detail that each model includes for particular tasks examined.

Andrews and Goodson selected fourteen instructional development tasks with very broad descriptions. For example, “Task 11: Need” includes the following subtasks: “Assessment of need, problem identification, occupational analysis, competence, or training requirements” (p. 5). There is no indication in their review as to which of the several itemized subtasks are the one(s) actually included in a particular model. Stamas identified sixteen tasks common to at least three of the models he reviewed, defining these tasks somewhat more narrow.

Andrews and Goodson evaluated all of the 40 models they reviewed across all fourteen tasks. After describing 23 models individually, Stamas reported a similar, though more specific, across-models analysis. Both reviews employed a models-by-tasks matrix to report the major tasks included in each of the models reviewed: the scope of these analyses is at the task level. No information is provided about how many subtasks are included in each task included in a model, nor about how thoroughly the task is explained in a model.

New Analysis Process: Task and Subtask Levels

A process has been devised which, when conducted by an experienced instructional developer, yields a more precise comparative analysis of models at the subtask level. The steps of this “micro-analysis” are presented in this article. Several terms are introduced in order to describe the analysis process and its results; each is defined when first used. Although the terms may at first appear to be somewhat lengthy and cumbersome, they are both functional and descriptive.

Subtask and Task Identification

The first part of the new analysis process involves creating a list of major tasks, with the specific activities contributing to each major task identified as subtasks of that task. This is done in three steps:

1. Determine an extensive list of models applicable to the creation of elementary and secondary instructional materials (or the context of choice).
2. Use these models to generate an exhaustive list of specific instructional development activities.
3. Group related activities into major tasks, with the constituent activities designated as subtasks of the major tasks.

In order to determine what development tasks are recommended by model builders, an extensive literature review was conducted. As a beginning, those models already reviewed by Andrews and Goodson (1980), Diamond (1985), Gentry (1984), Gustafson (1981), Reigeluth (1983), Stamas (1973), and Twelker, Urbach, and Buck (1972) were considered. Additional potential models were identified through a computer search of the ERIC and Psychological Abstracts data bases.

Examination of 30 models selected as relevant to the development of school materials resulted in the identification of 67 basic instructional development subtasks (see Appendix I). These subtasks have been grouped into sets of related activities to form twelve major tasks:

<table>
<thead>
<tr>
<th>Task</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Task 1</td>
<td>Needs assessment</td>
</tr>
<tr>
<td>Task 2</td>
<td>Goals and objectives specification</td>
</tr>
<tr>
<td>Task 3</td>
<td>Resource and constraint analysis</td>
</tr>
<tr>
<td>Task 4</td>
<td>Target population description</td>
</tr>
<tr>
<td>Task 5</td>
<td>Task analysis</td>
</tr>
</tbody>
</table>
In carrying out this process, the major tasks and subtasks of a comprehensive development model (Briggs, 1977) were used initially, then additional models were reviewed. If new subtasks were identified, the subtask lists were extended; if necessary, the major tasks were also redefined. A flowchart of this iterative tabulation and grouping procedure is included as Figure 2.

Clearly, the model selection process will influence which subtasks are identified for each major task. However, in this case, the grouping of subtasks into major tasks stabilized quite early in the analysis process, so it is unlikely that the list of major tasks has been affected by the selection process.

The twelve major tasks are at about the same level of generality as those identified by Andrews and Goodson (1980) and by Stamas (1979). At the overall task level, there is considerable similarity between the major tasks defined here and those identified in the other reviews. The instructional development tasks of this analysis and those enumerated by Andrews and Goodson and by Stamas are compared in Table 1. These comparisons are only general, because the subtasks included in the major tasks of the earlier analyses do not correspond closely with each other or with those determined here.

**Task-Level Analysis of Models**

The model builders' stated purposes determine the major tasks that are analyzed for each model. For example, if a model builder did not outline a procedure for a given major task, such as "Task 1: Needs Assessment," then the model is not analyzed for that task. Therefore, models are first analyzed to determine which major tasks they include.

**Subtask-Level Analysis**

The second level of analysis applied to the models determines which sub-
**TABLE 1**

Major Development Tasks Defined in This Analysis Compared with Tasks Identified in Previous Analyses

<table>
<thead>
<tr>
<th>Major Tasks</th>
<th>Andrews &amp; Goodson (1980)(^1)</th>
<th>Stamos (1973)(^2)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 Needs Assessment</td>
<td>11 Need</td>
<td>4 Identify Problem</td>
</tr>
<tr>
<td></td>
<td>12 Alternatives</td>
<td></td>
</tr>
<tr>
<td>2 Goals &amp; Objectives</td>
<td>1 Outcomes</td>
<td>1 Broad Instructional Goals</td>
</tr>
<tr>
<td>Specification</td>
<td></td>
<td>6 Specification of Behavioral Objectives</td>
</tr>
<tr>
<td>3 Resource &amp; Constraint</td>
<td>13 Constraints</td>
<td></td>
</tr>
<tr>
<td>Analysis</td>
<td>14 Cost</td>
<td></td>
</tr>
<tr>
<td>4 Target Population</td>
<td>5 Learner Attributes</td>
<td>5 Pre-Assessment of Entry Skills</td>
</tr>
<tr>
<td>Description</td>
<td>3 Analysis</td>
<td>2 Collect Data</td>
</tr>
<tr>
<td>5 Task Analysis</td>
<td></td>
<td>7 Enabling Objectives</td>
</tr>
<tr>
<td>6 Test Construction</td>
<td>2 Tests</td>
<td>8 Task Analysis</td>
</tr>
<tr>
<td>7 Instructional Sequencing</td>
<td>4 Sequencing</td>
<td>6 Specification of Performance Tests</td>
</tr>
<tr>
<td>8 Instructional Planning</td>
<td>6 Strategy</td>
<td>10 Review/Revise Instructional Content</td>
</tr>
<tr>
<td>9 Media Selection</td>
<td>7 Media</td>
<td>9 Analyze Setting</td>
</tr>
<tr>
<td>10 Instructional Materials</td>
<td>8 Development</td>
<td>14 Design Teaching/Learning Activities</td>
</tr>
<tr>
<td>Specification</td>
<td></td>
<td></td>
</tr>
<tr>
<td>11 Materials Production</td>
<td>8 Development</td>
<td>12 Select Design Format</td>
</tr>
<tr>
<td>12 Formative Evaluation</td>
<td>9 Tryout/Revision</td>
<td>13 Construct Prototype</td>
</tr>
<tr>
<td></td>
<td></td>
<td>11 Technical &amp;</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Communications Review</td>
</tr>
<tr>
<td></td>
<td></td>
<td>13 Test Prototype</td>
</tr>
</tbody>
</table>

\(^1\)In addition, Andrews and Goodson identified Task 20, Install/Maintain.

\(^2\)Stamos also identified Task 2, Organize Management; Task 15, Support Services; and Task 16, Implement/Evaluate/Revise/Recycle/Feedback Loop.

Tasks of each relevant major task are included in the model (comprehensiveness for that task), and which subtasks are explained thoroughly (operational level for that task).

These analyses are sharpened by the use of a weighting factor for each sub-task. A subtask weight within each major task is simply the number of relevant models that include the subtask. The assignment of weights is not done at the task level, because all of the major development tasks are essential, at least to some degree. At the subtask level, different but equally valid methods for carrying out a task are presented by different model builders. Subtask weights demonstrate the overall acceptance of each of these methods by instructional development theorists.

**Comprehensiveness of Task Coverage**

For each major task, a weighted comprehensiveness rating is then assigned to each relevant model, as follows:

\[
\text{Comprehensiveness rating} = \frac{\text{sum of weights of included subtasks}}{\text{sum of weights of all subtasks in that task}}
\]

and expressed as a percent. (Alternatively, one could just count the subtasks and determine a simple proportion. However, that approach would give no indication of the relative acceptance, or "importance," of the subtasks included.)

**Operational Level of Task Coverage**

A model is considered to be "operational" with respect to a given subtask if it supplies enough detail to teach that subtask, or if a student (or a person with little knowledge about instructional development procedures) could carry out that task using only the information included in that model. Thus, a model in which the process for a given subtask is described very briefly, or in which only references to other sources are given, is not operational for that subtask.
In the models analysis process, for each subtask included in a model, an informed yes/no judgment is made by one or more experienced instructional developers as to whether the model is operational for that subtask. A weighted operational level rating (expressed as a percent) is then assigned to each relevant model in each major task, as follows:

\[
\frac{\text{sum of weights of operational subtasks}}{\text{sum of weights of all subtasks in that task}}
\]

**Full-Model Analysis**

Although the comprehensiveness and operational level ratings of a model for each task addressed are important data, questions remain about how comprehensive or operational that model is for all of the tasks it addresses. To answer these questions, it is necessary to display the individual ratings in larger matrices and derive some broader ratings from the subtask- and task-level analyses.

**Mean Relative Comprehensiveness.** Simply counting the number of tasks addressed would give, for each model, a measure of how "broadly comprehensive" the model is; that is, the approach taken by Andrews and Goodson (1980) and by Stamas (1975). However, task-by-task comprehensiveness ratings can be used to provide a finer measure for those tasks included in each model.

Taken together, the task-by-task comprehensiveness ratings show the overall comprehensiveness of each model relative to the model-builder's intended purpose. A mean relative comprehensiveness rating is derived as follows:

\[
\frac{\text{sum of individual comprehensiveness ratings}}{\text{number of tasks included}}
\]

**Mean Relative Operational Level.** The task-by-task operational level ratings show the overall operational level of each model relative to the model-builder's intended purpose. A mean relative operational level rating is derived as follows:

\[
\frac{\text{sum of individual operational level ratings}}{\text{number of tasks included}}
\]

**Selecting Reference Models.** Instructional developers can use the data provided by this analysis process to select one or more reference models suitable for their needs.

In order to identify an "optimal" instructional development model, the most comprehensive model(s) for each major task could be determined. The complete set of comprehensive models so identified could be considered to represent an overall, eclectic, comprehensive "model" for all major tasks. Alternatively, a single "broadly comprehensive" model could be chosen, based on both the number of tasks addressed and the relative mean comprehensiveness rating. For some tasks, any model would, in all likelihood, have to be supplemented by additional models.

Similarly, the most operational model(s) for each major task could be identified. The complete set of operational models so determined could represent an overall, eclectic, operational "model" for all major tasks. Selecting a "broadly operational" model would require consideration of both the number of tasks addressed and the relative mean operational level rating.

**An Illustration of the Analysis Process**

Four instructional development models (Briggs, 1977; Control Data Corporation, 1979; Dick & Carey, 1978; Gagné & Briggs, 1979) and one instructional design model (Reigeluth & Stein, 1983) illustrate how the analysis process is used.

**Task-Level Analysis of Models.** The major tasks addressed by each of the five selected models are identified in Table 2. This simple matrix is very similar to those created by Andrews and Goodson (1980) and by Stamas (1975).

In this analysis process, each model is analyzed only for the tasks it addresses to some degree. In other words, not all models are considered for each task. Therefore, the number of models contributing to the data for each task varies. As shown in the far right column of Table 2, the number of models addressing a given task ranges from one (Task 11) to five (Tasks 5, 7, 8). Because it is a task that all five of these models include, "Task 7: Instructional Sequencing" will be used to illustrate the subtask-level portions of the analysis process.

**Subtask Weights.** For Task 7, those subtasks included by each of the five models are shown by open bullets in Table 3. Each of the five subtasks, 7.1 through 7.5, is included in at least one, and at most five, of the models. The number of models including each subtask gives the subtask weights shown in the far right column of Table 3.

**Comprehensiveness.** Summing the weights of included Subtasks 7.1 through 7.5 for each model, and dividing by the sum of the weights of all subtasks in Task 7 (14), gives the comprehensiveness ratings in the bottom row of Table 3. In this example, Dick and Carey (1978), at 93%, is the most comprehensive model for Task 7. A different model might be the most comprehensive for a different task.
### TABLE 2
Major Tasks Addressed by Selected Models

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Needs assessment</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td></td>
<td>3</td>
</tr>
<tr>
<td>2. Goals &amp; objectives specification</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td></td>
<td>4</td>
</tr>
<tr>
<td>3. Resource &amp; constraint analysis</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td></td>
<td>3</td>
</tr>
<tr>
<td>4. Target population description</td>
<td></td>
<td></td>
<td>X</td>
<td></td>
<td></td>
<td>2</td>
</tr>
<tr>
<td>5. Task analysis</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td></td>
<td>5</td>
</tr>
<tr>
<td>6. Test construction</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td></td>
<td>4</td>
</tr>
<tr>
<td>7. Instructional sequencing</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td></td>
<td>5</td>
</tr>
<tr>
<td>8. Instructional planning</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td></td>
<td>5</td>
</tr>
<tr>
<td>9. Media selection</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td></td>
<td>4</td>
</tr>
<tr>
<td>10. Instructional materials specification</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td></td>
<td>4</td>
</tr>
<tr>
<td>11. Materials production</td>
<td></td>
<td></td>
<td>X</td>
<td></td>
<td></td>
<td>1</td>
</tr>
<tr>
<td>12. Formative evaluation</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td></td>
<td>4</td>
</tr>
</tbody>
</table>

### TABLE 3
Comprehensiveness Ratings of Five Models for Task 7 Instructional Sequencing, Taking Subtask Weights into Account

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>7.1 Verify enabling objectives</td>
<td></td>
<td></td>
<td>o</td>
<td>o</td>
<td>o</td>
<td>3</td>
</tr>
<tr>
<td>7.2 Select organizing content</td>
<td></td>
<td></td>
<td>o</td>
<td>o</td>
<td>o</td>
<td>1</td>
</tr>
<tr>
<td>7.3 Determine overall teaching order</td>
<td></td>
<td></td>
<td>o</td>
<td>o</td>
<td>o</td>
<td>5</td>
</tr>
<tr>
<td>7.4 Determine “size” of lessons</td>
<td></td>
<td></td>
<td>o</td>
<td>o</td>
<td>o</td>
<td>3</td>
</tr>
<tr>
<td>7.5 Validate instructional sequence</td>
<td></td>
<td></td>
<td>o</td>
<td>o</td>
<td></td>
<td>2</td>
</tr>
</tbody>
</table>

COMPREHENSIVENESS RATING (%): (10/14) 71 (8/14) 57 (13/14) 93 (8/14) 57 (9/14) 64 (14/14) 100

### TABLE 4
Operational Levels of Models for Task 7 Instructional Sequencing, Taking Subtask Weights into Account

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>7.1 Verify enabling objectives</td>
<td></td>
<td></td>
<td>o</td>
<td>o</td>
<td>o</td>
<td>3</td>
</tr>
<tr>
<td>7.2 Select organizing content</td>
<td></td>
<td></td>
<td>o</td>
<td>o</td>
<td>o</td>
<td>1</td>
</tr>
<tr>
<td>7.3 Determine overall teaching order</td>
<td></td>
<td></td>
<td>o</td>
<td>o</td>
<td>o</td>
<td>5</td>
</tr>
<tr>
<td>7.4 Determine “size” of lessons</td>
<td></td>
<td></td>
<td>o</td>
<td>o</td>
<td>o</td>
<td>3</td>
</tr>
<tr>
<td>7.5 Validate instructional sequence</td>
<td></td>
<td></td>
<td>o</td>
<td>o</td>
<td></td>
<td>2</td>
</tr>
</tbody>
</table>

OPERATIONAL LEVEL RATING (%): (8/14) 57 (0/14) 0 (3/14) 21 (5/14) 36 (8/14) 57 (14/14) 100
Operational Level. In Table 4, the subtasks of Task 7 for which descriptions in the models are operational are indicated by solid bullets; subtasks for which the descriptions are not operational are indicated by open bullets. When the weights of operational subtasks are summed for each model and compared to the sum of the weights of all subtasks in Task 7, the operational level ratings in the bottom row of Table 4 result. In this example, Briggs (1977) and Reigeluth and Stein (1983) are the most operational models for Task 7, both at 57%. A different model might be the most operational for a different task.

Full-Model Analysis. For simplicity, the preceding illustrations have focused on the analysis of five models for a single task. However, because very few of the five selected models include certain tasks (refer to Table 2, Tasks 1, 3, 4, and 11), it is necessary to consider a greater number of models in order to illustrate meaningful across-tasks data. Therefore, the data for twenty instructional development models (see Appendix II) have been used to compute the mean relative comprehensiveness and operational level ratings.

Using a greater number of models increases both individual subtask weights and the sum of subtask weights for a task. However, the resulting task-by-task ratings for a given model differ very little. In Tables 5 and 6, the ratings for the individual models for Task 7 differ only slightly from those in the preceding illustrations (compare with Tables 3 and 4). (The total number of models contributing to the ratings for each task is displayed in the far right columns of Tables 5 and 6.)

In Table 5 are displayed the comprehensiveness ratings for each task addressed by the five selected models. The ratings were determined using all relevant data for twenty instructional development models, not just the five selected for reporting here. The last two rows of Table 5 show the number of tasks included in each model and the mean relative comprehensiveness rating for each model. Of these five models, the model with the highest mean relative comprehensiveness rating is Dick and Carey (1978), with a rating of 89%. Note that, although the Dick and Carey model includes just nine of the twelve tasks, it is highly comprehensive (rating ≥ 80%) for seven tasks and the most comprehensive of these five models for five tasks. Together, the Gagné and Briggs (1979) and Control Data Corporation (1979) models provide highly comprehensive coverage of the three tasks missing from Dick and Carey (Tasks 1, 3, 11) and could also be used to amplify the treatment of Tasks 2, 9, 10, and 12.

In Table 6 are displayed the operational level ratings for each task addressed by the five selected models. These ratings were determined using all relevant data for twenty instructional development models. The last two rows of Table 6 show the number of tasks addressed by each model and the mean relative operational level rating for each model. Of these five models, the model with the highest mean relative operational level rating is Briggs (1977), with a rating of 50%. Note that, although Briggs includes just ten of the twelve tasks, it is fairly operational (rating ≥ 50%) for seven tasks and the most operational of the five models for five tasks.

In selecting a single reference model from among the five illustrated here, an experienced instructional developer might choose the Control Data model (1979) because it addresses all twelve tasks and is fairly comprehensive (mean relative comprehensiveness rating 77%). Such a practitioner's experience could be sufficient to “fill the gaps” suggested by the model’s operational level ratings. However, that model’s low mean relative operational

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Needs assessment</td>
<td>60</td>
<td>57</td>
<td>77</td>
<td>8</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2. Goals and objectives specification</td>
<td>94</td>
<td>88</td>
<td>66</td>
<td>94</td>
<td>13</td>
<td></td>
</tr>
<tr>
<td>3. Resource and constraint analysis</td>
<td>88</td>
<td>88</td>
<td>83</td>
<td>83</td>
<td>5</td>
<td></td>
</tr>
<tr>
<td>4. Target population description</td>
<td>75</td>
<td>100</td>
<td>75</td>
<td>75</td>
<td>7</td>
<td></td>
</tr>
<tr>
<td>5. Task analysis</td>
<td>100</td>
<td>79</td>
<td>100</td>
<td>83</td>
<td>83</td>
<td>14</td>
</tr>
<tr>
<td>6. Test construction</td>
<td>89</td>
<td>75</td>
<td>100</td>
<td>75</td>
<td>11</td>
<td></td>
</tr>
<tr>
<td>7. Instructional sequencing</td>
<td>81</td>
<td>67</td>
<td>95</td>
<td>62</td>
<td>67</td>
<td></td>
</tr>
<tr>
<td>8. Instructional planning</td>
<td>63</td>
<td>51</td>
<td>77</td>
<td>75</td>
<td>35</td>
<td>13</td>
</tr>
<tr>
<td>9. Media selection</td>
<td>75</td>
<td>75</td>
<td>91</td>
<td>100</td>
<td>14</td>
<td></td>
</tr>
<tr>
<td>10. Instructional materials specification</td>
<td>68</td>
<td>96</td>
<td>84</td>
<td>68</td>
<td>11</td>
<td></td>
</tr>
<tr>
<td>11. Materials production</td>
<td>100</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>12. Formative evaluation</td>
<td>68</td>
<td>68</td>
<td>90</td>
<td>100</td>
<td>9</td>
<td></td>
</tr>
<tr>
<td>NO. OF TASKS</td>
<td>10</td>
<td>12</td>
<td>9</td>
<td>10</td>
<td>3</td>
<td></td>
</tr>
<tr>
<td>MEAN RELATIVE COMPREHENSIVENESS (%)</td>
<td>79</td>
<td>77</td>
<td>89</td>
<td>80</td>
<td>62</td>
<td></td>
</tr>
</tbody>
</table>
level rating (8%) limits its usefulness to students and other inexperienced instructional developers. For such novices, either the Briggs model, with mean relative ratings of 79% and 50% across ten tasks, or the Dick and Carey model, with ratings of 89% and 45% across nine tasks, would appear to be more helpful.

Concluding Remarks

In this article a process for analyzing prescriptive models for the development of instructional materials has been described. The process has four steps:

1. Group related subtasks into major tasks.
2. Analyze each selected model relative only to the major tasks the model actually includes.
3. Within each major task, determine weights for the subtasks, as a measure of the "importance" assigned to each subtask by the field.
4. For each major task, analyze relevant models for comprehensiveness and operational level, using the sum of the subtask weights as the base.

This model-analysis process is iterative and cybernetic. As they are published, new models can be analyzed and the resulting data integrated with previous results. At the same time, the parameters dependent on subtask weights can be refined by input from those new models.

The process for analyzing models developed in this article can be applied to the analysis of instructional development models intended for other contexts, such as military training. The process may also prove to be of value in analyzing and comparing models that are intended for quite different purposes, such as courseware evaluation.

Thanks are extended to Dr. Donald P. Ely and Dr. Sidney S. Miez of Syracuse University and to Dr. Jack E. Forbes of Purdue University for their contributions to this effort.

References

APPENDIX I. The Tasks and Subtasks of Instructional Development (in brief)

Task 1: Needs Assessment. The task of needs assessment involves determining goals, identifying discrepancies between goals and the status quo, and establishing priorities for action.

1.1 Identify and rank a range of possible goals.
1.2 Identify discrepancies between expected and actual performance.
1.3 Analyze consequences of discrepancies.
1.4 Determine possible solution areas.
1.5 Set priorities for action.
1.6 Select intervention.

Task 2: Goals and Objectives Specification. The task of determining instructional goals and objectives involves generating increasingly specific objectives from more general objectives.

2.1 Define overall instructional goals.
2.2 Specify end-of-course objectives.
2.3 Specify unit objectives.
2.4 Write terminal behavioral (performance) objectives.
2.5 Draw instructional map.
2.6 Determine objectives “fit” within curriculum.

Task 3: Resource and Constraint Analysis. This task involves determining whether a cost-effective development effort can be undertaken.

3.1 Assess resources available.
3.2 Analyze existing constraints.
3.3 Plan constraint removal.
3.4 Make go/no-go decision.
3.5 Plan resource use.

Task 4: Target Population Description. This task involves determining the characteristics of the target population, or students, so that instructional materials can be prepared to suit the learners’ needs.

4.1 Determine general characteristics.
4.2 Determine aptitude, ability, and skill levels.
4.3 Determine attitude and motivational characteristics.

Task 5: Task Analysis. Task analysis provides a conceptualization for the instructional design and useful guidance for the writing of assessment devices.

5.1 Identify and classify tasks/content to be learned.
5.2 Conduct information-processing/content analysis.
5.3 Conduct learning task analysis.

5.4 Draw learning map.
5.5 Define entry behaviors.
5.6 Validate objectives.

Task 6: Test Construction. The student’s performance level on each prerequisite or terminal objective is determined in order to monitor each learner’s progress and thus prevent failures and minimize remedial instruction.

6.1 Specify administrative details/assessment system.
6.2 Specify appropriate test characteristics for each prerequisite or terminal objective.
6.3 Construct and review test items.
6.4 Determine test validity and reliability.
6.5 Try out the test and revise it.

Task 7: Instructional Sequencing. The general sequencing of instruction among terminal objectives is the aim of Task 2. Task 7 involves the sequencing of instruction among enabling objectives.

7.1 Verify enabling objectives for each terminal objective.
7.2 Select the organizing content.
7.3 Determine overall teaching order of enabling objectives/content.
7.4 Determine “size” of lessons.
7.5 Validate instructional sequence.

Task 8: Instructional Planning. Instructional planning includes specifying instructional events, or teaching steps, for each enabling objective. Not all teaching steps must be built into the instructional materials. Some may be provided by the teacher, others by the student.

8.1 Identify content.
8.2 Plan pacing and grouping of instruction.
8.3 Identify options for instructional methods.
8.4 Select instructional methods.
8.5 Plan pre-instructional activities.
8.6 Plan presentation of new content.
8.7 Plan practice with feedback.
8.8 Plan performance assessment.
8.9 Plan for retention and transfer.
8.10 Specify conditions of learning.

Task 9: Media Selection. This task involves selecting appropriate media by considering task variables, learner variables, the assumed learning environment, Continued
APPENDIX I. (Continued)
the assumed product development environment, the
economy and culture, and practical factors.

9.1. Determine fundamental delivery methods.
9.2. Specify stimulus characteristics.
9.3. Specify response characteristics.
9.4. Identify potential media.
9.5. Make final media selection.

Task 10: Instructional Materials Specification. This task involves preparing manuscripts, scripts, sketches, and storyboards for the instructional materials.

10.1. Prepare instructional standards.
10.2. Review existing materials.
10.3. Write prescriptions.
10.4. Review and revise prescriptions.

Task 11: Materials Production. This task involves
producing first the prototype, then the final text, audio,
and visual materials that will make up the instructional
package.

11.1. Draft and review the learning activities.
11.2. Edit and revise draft materials.
11.3. Produce audio-visual materials.
11.4. Assemble course materials.
11.5. Revise on basis of field trial results.
11.6. Produce text materials.

Task 12: Formative Evaluation. This task involves
three phases of product validation, after each of which
the instructional materials are revised.

12.1. Plan materials evaluation system.
12.2. Describe learning environment.
12.3. Conduct one-to-one evaluation.
12.4. Conduct small-group evaluation.
12.5. Train field trial teachers.
12.6. Conduct field trial evaluation.

APPENDIX II. Models Used in Computing Mean Relative Comprehensiveness and Operational Level Ratings.


Teachers' Beliefs About Instructional Computing: Implications for Instructional Designers

Nancy Nelson Knupfer
Assistant Professor
Arizona State University
Educational Media and Computers
Tempe, AZ 85287-0111

Abstract. This article reports selected results of a study in which teachers' perceptions, opinions, and attitudes about instructional computing were examined. Implications about equitable access to computers in public schools are described. The data were gathered via a questionnaire mailed to 510 sixth-grade teachers in K-6 structured public schools. Significant findings concerning the nature of teachers' thoughts and experiences with instructional computing, and the potential effect of those factors on students' access to computers, are reported. As a needs assessment, this study provides useful information to instructional designers about how teachers perceive the computer and its use in their classrooms. Based upon the study, factors to consider when designing computer-based instruction for implementation in schools are suggested.

Computer technology is increasingly available for both home and school use, and the demands of parents, industry, and students have caused schools to purchase ever larger numbers of microcomputers (Becker, 1986a; Marapodi, 1984). However, this rapid acquisition of new equipment has not necessarily improved education. Indeed, the novelty of the computer stands in immediate contrast to the traditional attitudes and teaching methods and the rigid bureaucracy that survived the initial computer invasion of the 1960s (Sheingold, Kane, Endrews, & Billings, 1981). Furthermore, this immense growth in educational computing has been accompanied by inequitable practices both in the curriculum and in the students' access to the machines.

Several studies show that inequity exists in students' access to computers in schools (Anderson, Welch, & Harris, 1984; Becker, 1982, 1985, 1986b; Schubert & Bakke, 1984; Sheingold, et al., 1981) and that inequities frequently arise early in the implementation process (Emocha, 1984; Lacina, 1984). Edwards (1984) in particular distinguishes between quantitative and qualitative inequities. Quantitative inequity results from shortages of resources. Qualitative inequity involves intangible attitudes and institutional biases that pose greater long-term threats to equal access and use.

The distinction is in many ways less important when qualitative and quantitative factors combine to create inequities that cannot be corrected by addressing a single cause. Both kinds of factors lead to inequitable decision making: the school or the teacher responds by using unfair priorities that limit access. Schools often use computers selectively, scheduling access to the advantage of certain populations (i.e., boys, affluent or gifted students), over others (Becker, 1984, 1985, 1986b; Schubert & Bakke, 1984). No matter how distant, varying, or indeterminate their source, inequities converge in the classroom and are closely related to the way teachers think about and use computers.

As the most salient research confirms, the teachers' particular preconceptions are crucial to the course of innovation and change, including the process of bringing products of instructional systems design into the schools. Even though many such products are adopted by the most farsighted, enlightened school administrations, they are doomed if classroom teachers don't perceive them as meaningful within the current classroom environment (Cuban, 1986; Fullan, 1982; Weinschank, Trumbull, & Daly, 1983).

When implementing the material and procedural products of instructional systems design within the schools, it is imperative to consider the importance of both constancy and change. According to Cuban (1986), teachers work within a realm of embedded policies and work routines which must adjust to any proposed change. He points out the positive aspects of stability and craft within instruction, and notes that because "teachers are the gatekeepers for instructional technology" (p. 37), they must be consulted and involved in the introduction of any new product, including educational computing. Thus, any change in teaching practice requires the investigation and assessment of the teacher's viewpoint at the succeeding stages of planning and implementation, as well as a clear definition of what is to be implemented.
...any change in teaching practice requires the investigation and assessment of the teacher's viewpoint at the succeeding stages of planning and implementation, as well as a clear definition of what is to be implemented.

Further, the importance of how a new product is implemented is much more important than what is implemented (Berman & McLaughlin, 1976). Accordingly, the more closely computer implementation is attuned to the existing rhythms of school life, the better the chance of meaningful computer use for all students (Cuban, 1986; Fullan, 1982). Computer-based products will not live up to their full potential in the public schools until teachers accept them as a valuable part of the school curriculum, and have enough training and equipment to properly implement them.

the schools, these findings provide instructional designers with useful information.

Method

The data for this study were gathered through a mailed survey. Descriptive statistics were compiled from frequency distributions and measures of central tendency. Statistical analyses were conducted with contingency tables using the chi-square statistic, t tests, ANOVA, and correlations. The alpha level for significance was set at .05.

Purpose

The purpose of this study was to examine teacher perceptions, opinions, and attitudes about instructional computing and to draw conclusions about the quantity and quality of equitable access to computers in public elementary schools. Sixth-grade teachers were surveyed about the status of instructional computing, and their responses to instructional computing curricula were examined. Findings about selected aspects of students' access to computers and teachers' training and beliefs regarding instructional computing which may affect equity of students' computing experiences in schools are reported. Because good instructional design includes a responsibility to promote educational equity in

Because good instructional design includes a responsibility to promote educational equity in the schools, these findings provide instructional designers with useful information.
of microcomputer usage, the composition of groups receiving instruction, the various instructional applications of computing, the amount of teacher training, and the equity of access among students.

Results

The Sample

T tests and chi-square analyses on several variables revealed no statistically significant differences between the respondent and non-respondent groups except for the number of years of teaching experience at the current school; non-respondents averaged three years more teaching than respondents.

Inequities in Wisconsin's Educational Computing

Material Inequities. Of the 95 school districts represented by respondents, all had at least one computer, and the majority of schools (238) within those districts had their own computers. Only 17 of the 369 teacher respondents reported no access to computers. Those 17 teachers represented 14 schools which were spread throughout several districts, cooperative educational service areas (CESAs) and counties. These results indicate that although all school districts had computers, the distribution of computing resources within districts varied.

Of the 238 schools with computers, 55% had five or fewer computers, 30% had between six and ten machines, and 15% had more than ten machines (see Figure 1). The median number of computers per school (5) and the ratio of students per computer (38:1) in Wisconsin were similar to the figures revealed by a nationwide sample (6 and 60:1, respectively), conducted by Henry Becker (1986a) during the same report period.

Spatial Inequities. Several factors—location, length of time computers could be used, how often classroom computers were used, and how many hours they were actually turned on during a day of use—highlighted inequity in access to computers among sixth-grade students. The large differences in the total time each student had access to a computer in the classroom depended upon the total amount of time allocated for specific classrooms to share the computing resources; classroom access ranged from no time to the entire school year. Furthermore, different patterns of computer access reported by teachers indicated that students' access to computers might very well depend upon the attitude of the particular classroom teacher.

Of the 352 teachers with computers in their schools, about 80% claimed that fewer than half of the students used computers on a regular, weekly basis. The major reasons teachers gave for "less than weekly use" (see Figure 2) were divided among problems concerning remote location of computers (61%), time constraints of the main curriculum (39%), and lack of teacher training (58%). Following in order of importance were lack of computers (19%), funding restrictions to specific student groups (22%), lack of necessary software (10%), undeveloped program of study (7%), funding limitations to specific curricular areas (6%), and lack of student interest (3%).

What Teachers Think

The Benefits of Instructional Computing. The teachers' attitudes and opinions about the value of educational comput-

Figure 1. Number of computers per school.

Figure 2. Reasons for irregular use of computers.
ing in their schools varied widely, and reflected a general confusion about the purposes of instructional computing. Most teachers believed that instructional computing had some benefit, but they were divided about what those benefits were and how they applied to the purposes of education.

When asked to list one or two most important benefits of instructional computing, 40% of the teachers claimed that student motivation was the predominant benefit of instructional computing (see Figure 3). They specifically commented that instructional computing added interest, posed a challenge, could be used as a reward, and offered variety in the school day. Other benefits attributed to computer use included: preparation for survival in the computer age and for life in general (19%), increased learning (11%), preparation for the job market of the future (10%), teaching higher-order thinking or problem-solving skills (6%), and variety in classroom activities (3%). Approximately 7% of the teachers claimed that there was absolutely nothing good about instructional computing.

Worst Aspects and Major Problems. The teacher's complaints about the worst aspects of instructional computing focused on material, managerial, and training problems (see Figure 4). Heading the list was the shortage of computers and/or software (36%). Within this category, some teachers mentioned that lack of space prevented any increase in inventory. Similar comments referred to both the unjustifiable expense of computers relative to other important curricular priorities and the poor quality of software (10.9% each). Teachers were concerned that commercially available software did not match their curricular requirements for skill level and content.

Major managerial concerns included time constraints of the daily curriculum (25.3%) and time limitations in teachers' schedules (9.6%). Together, these two categories comprise a larger category of general time constraints, becoming the most predominant single problem, even greater than lack of program direction or teacher training. Time limitations in the teachers' schedules precluded planning for instructional computing, reviewing software, and both long-range and immediate implementation of instructional computing into the existing curriculum.

About 20% of the teachers believed that inadequate teacher training was the worst aspect of instructional computing, and another 17.7% faulted the lack of any program direction. Approximately 22% of the teachers questioned the value of educational computing. They were aware that their fellow teachers did not care about or were afraid of instructional computing, or believed that the computing objectives were unrealistic. Questionable educational value was mentioned both by those who had and who had not used instructional computing.

When asked to list one or two major problems, 275 teachers who had computers in their schools blamed lack of time (48%), lack of computers (37%), and lack of teacher training (33%) (see Figure 5). These responses are consistent with the top three worst aspects reported earlier. The emphasis on such management issues as supervision and grading procedures (24%) suggests the extent to which teachers who use computers are still struggling to fit them into their normal routines. Lack of program objectives (23%) and poor-quality software (15%) also were frequently indicated. Smaller percentages of teachers who had computers in their schools believed that remote location of computers (8%), low teacher interest (6%), and poor administrative support (5%) were major problems.
teachers. Teachers do not view the computer as a valuable component of a standard curriculum that should be made accessible to all students at some level.

At least the criterion of remedial instruction (i.e., drill and practice for students who needed to improve basic skills) has some educational validity. The danger here is more likely to lie in a steady diet of drill-and-practice programs which rarely direct the student’s curiosity about the computer to further development of higher-order skills.

Teachers’ Training for Instructional Computing

Teachers frequently reported inadequate training and deferred computer access to their more competent colleagues. Training requirements for teachers existed in only 14% of the schools and were not consistent within districts. The median amount of inservice training time received by all teachers surveyed was three hours. The median amount of college or university credits, vocational school classes, community education, or training from other sources in computer use was zero. A few teachers (1%) had many hours of inservice training, but the large majority had none at all. The teachers who had participated in formal classes generally reported one university-level course in educational computing.

To determine whether teachers’ age or years of teaching experience were correlated with amount of computer training, analysis of variance tests were performed on four training categories: inservice hours, college computer credits, semesters of technical school, and hours of local private training. The analyses of variance by both age and years of experience showed significant differences only for the number of college computer credits earned. Scheffe multiple-range post hoc tests of each category revealed a significant difference between the youngest and oldest age groups (groups one and four in Table 1), and between the least experienced and most experienced teachers (groups one and four in Table 2). Younger teachers had a mean of 1.74 computer credits while older teachers had a mean of 4.8 credits. The least experienced teachers had a mean of 2.11 credits, while the most experi-

Figure 5. Major problems with instructional computing.

![Figure 5](image1)

Figure 6. Criteria suggested for determining access to computers.

![Figure 6](image2)

Providing Student Access to Computers.
Confusion of attitudes and opinions about the goals, value, and general validity of educational computing was particularly apparent when the teachers were asked to devise criteria for access to a limited number of computers. This question was directed to all teachers, regardless of whether or not their schools currently had computers. Although “equal access” was the first choice of the largest number (37%) of teachers who responded to this question (see Figure 6), this criterion co-existed with views that hardly promote equal access, even in a rationing situation. For instance, the second most preferred criterion (27%) for determining students’ access was the ability of teachers to use computers. This criterion clearly would prevent some students from having access to a computer at all, especially those whose teachers believed that computers had no place in the schools.

Two other criteria, each mentioned by about 25% of the teachers, stressed the motivational or remedial purposes of computing. “Motivation” meant that the computer should be used to encourage good behavior. Better-behaved students would have a better chance of getting to the machine. This underscores the incoherence of instructional computing objectives, especially in the minds of those teachers who might conscientiously condemn inequity in theory but enforce it in practice.
enced teachers had a mean of .31 credits.

The variables of age and years of teaching experience may exhibit parallel patterns because, in the majority of cases, the number of years of teaching experience increases with age. The larger number of college computer credits shown for the younger and least experienced teachers may be due to more interest in computers, more willingness to accept innovations, or new requirements for instructional computer credits in teacher preparation programs. This study did not pinpoint the source, however. T tests performed on the variance categories of training to determine any differences in training by gender of the respondents showed no significant differences.

Training in Rural versus Urban Communities. Analysis of variance tests performed on each of the three main sources of teacher training (in-service hours, college computer credits, and semesters of technical school) between five community sizes (urban, suburban, medium city, small city, and rural) revealed a significant difference in hours of in-service training. A Scheffe post hoc analysis further identified a significant difference between urban and rural communities (see Table 3). The mean number of in-service training hours for urban communities was twice that of the rural communities.

The Relationship Between Attitudes and Training

When asked if they had enough skill to feel comfortable teaching various computer activities, teachers generally expressed either low levels of agreement or disagreement, based upon a Likert-type unipolar scale of 1 to 5 where 1 = strongly disagree and 5 = strongly agree. In descending order of strength of agreement, the teachers agreed most strongly that they were comfortable using computer software (M = 3.32). Teachers were uncertain about their comfort with teaching the rudimentary logic about how computers work (M = 3.03), teaching introductory BASIC programming (M = 2.88), and using Logo (M = 2.27). Teachers were uncomfortable teaching about advanced BASIC programming (M = 1.86), telecommunications (1.70), robotics (1.67), Pascal programming (1.64), data bases (1.58), and spreadsheets (1.53).

Although some teachers may be very comfortable with their training and current skill level, inadequate training surfaced throughout the questionnaire responses as a major problem. Teachers' responses concerning criteria for determining access, major problems, worst aspects, and reasons for nonuse were used to evaluate their beliefs about the adequacy of training. Teachers repeatedly complained of inadequate training throughout those four areas. About one-third (33%) of the teachers believed that inadequate training was a major problem, more

<table>
<thead>
<tr>
<th>TABLE 1</th>
<th>ANOVA for Training by Age</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>COLLEGE COMPUTER CREDITS</strong></td>
<td></td>
</tr>
<tr>
<td><strong>Source</strong></td>
<td>df</td>
</tr>
<tr>
<td>Between Groups</td>
<td>3</td>
</tr>
<tr>
<td>Within Groups</td>
<td>364</td>
</tr>
<tr>
<td>Total</td>
<td>367</td>
</tr>
</tbody>
</table>

Scheffe Multiple Range Posthoc Analysis = Significant Difference Between Group 1 and Group 4 at the .05 level

<table>
<thead>
<tr>
<th>Group</th>
<th>Age</th>
<th>N</th>
<th>Mean</th>
<th>SD</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>30 or younger</td>
<td>27</td>
<td>1.74</td>
<td>3.22</td>
</tr>
<tr>
<td>2</td>
<td>31-40</td>
<td>149</td>
<td>1.11</td>
<td>1.90</td>
</tr>
<tr>
<td>3</td>
<td>41-50</td>
<td>130</td>
<td>.80</td>
<td>1.70</td>
</tr>
<tr>
<td>4</td>
<td>51 or older</td>
<td>62</td>
<td>.48</td>
<td>1.18</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>TABLE 2</th>
<th>ANOVA for Training by Years of Teaching Experience</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>COLLEGE COMPUTER CREDITS</strong></td>
<td></td>
</tr>
<tr>
<td><strong>Source</strong></td>
<td>df</td>
</tr>
<tr>
<td>Between Groups</td>
<td>3</td>
</tr>
<tr>
<td>Within Groups</td>
<td>364</td>
</tr>
<tr>
<td>Total</td>
<td>367</td>
</tr>
</tbody>
</table>

Scheffe Multiple Range Posthoc Analysis = Significant Difference Between Group 1 and Group 4 at the .05 level

<table>
<thead>
<tr>
<th>Group</th>
<th>Years Experience</th>
<th>N</th>
<th>Mean</th>
<th>SD</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1-5</td>
<td>18</td>
<td>2.11</td>
<td>2.25</td>
</tr>
<tr>
<td>2</td>
<td>6-15</td>
<td>141</td>
<td>1.13</td>
<td>2.20</td>
</tr>
<tr>
<td>3</td>
<td>16-25</td>
<td>160</td>
<td>.84</td>
<td>1.63</td>
</tr>
<tr>
<td>4</td>
<td>26 or more</td>
<td>49</td>
<td>.31</td>
<td>1.10</td>
</tr>
</tbody>
</table>
TABLE 3
ANOVA for Teacher Training by Community Size

<table>
<thead>
<tr>
<th>Source</th>
<th>df</th>
<th>Sum of Squares</th>
<th>Mean Squares</th>
<th>F</th>
<th>F Prob</th>
</tr>
</thead>
<tbody>
<tr>
<td>Between Groups</td>
<td>4</td>
<td>1135.19</td>
<td>283.80</td>
<td>3.25</td>
<td>.01</td>
</tr>
<tr>
<td>Within Groups</td>
<td>362</td>
<td>31588.97</td>
<td>87.29</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Total</td>
<td>366</td>
<td>32734.16</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Scheppe Multiple Range Posthoc Analysis = Significant Difference Between Group 1 and Group 5 at the .05 level

<table>
<thead>
<tr>
<th>Group</th>
<th>Population</th>
<th>N</th>
<th>Mean</th>
<th>SD</th>
<th>Max</th>
</tr>
</thead>
<tbody>
<tr>
<td>1-Urban</td>
<td>50,000 or More</td>
<td>64</td>
<td>10.59</td>
<td>12.11</td>
<td>48</td>
</tr>
<tr>
<td>2-Suburban</td>
<td>Adjacent to Urban</td>
<td>58</td>
<td>6.93</td>
<td>10.03</td>
<td>60</td>
</tr>
<tr>
<td>3-Medium City</td>
<td>15,000-49,999</td>
<td>95</td>
<td>6.42</td>
<td>9.52</td>
<td>40</td>
</tr>
<tr>
<td>4-Small City</td>
<td>3,000-14,999</td>
<td>80</td>
<td>6.02</td>
<td>7.01</td>
<td>30</td>
</tr>
<tr>
<td>5-Rural</td>
<td>Less than 3,000</td>
<td>70</td>
<td>5.30</td>
<td>8.13</td>
<td>40</td>
</tr>
</tbody>
</table>

TABLE 4
t Tests on Training Between Teachers Who Do and Do Not Use Computers

<table>
<thead>
<tr>
<th>Variables</th>
<th>N</th>
<th>M</th>
<th>SD</th>
<th>df</th>
<th>t</th>
<th>P</th>
</tr>
</thead>
<tbody>
<tr>
<td>INSERVICE HOURS</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Nonuser</td>
<td>142</td>
<td>4.75</td>
<td>6.73</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>User</td>
<td>226</td>
<td>8.39</td>
<td>10.67</td>
<td>-4.02</td>
<td>.000</td>
<td></td>
</tr>
<tr>
<td>COLLEGE COMPUTER CREDIT</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Nonuser</td>
<td>142</td>
<td>.52</td>
<td>1.33</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>User</td>
<td>226</td>
<td>1.20</td>
<td>2.12</td>
<td>-3.83</td>
<td>.000</td>
<td></td>
</tr>
<tr>
<td>SEMESTERS TECHNICAL SCHOOL</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Nonuser</td>
<td>142</td>
<td>.01</td>
<td>.12</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>User</td>
<td>226</td>
<td>.10</td>
<td>.42</td>
<td>-2.99</td>
<td>.003</td>
<td></td>
</tr>
</tbody>
</table>

than a quarter (27%) indicated that those who had more training should have first access rights to computers, and one-fifth (21%) believed that inadequate teacher training was the worst aspect of instructional computing. Sixty-four percent of the teachers who had not used instructional computing indicated lack of training as the reason. Several teachers inserted unsolicited complaints about their administration's willingness to "jump on the bandwagon" and spend large sums of money on computing equipment and yet be unsupportive of teachers' training needs.

Training might boost a teacher's confidence and skill level, leading one to expect that better trained teachers would be more involved with instructional computing and have more positive attitudes about computers. T tests performed on the variable of training between teachers who used computers and those who did not showed significant differences in the amounts of inservice training, college computer credits, and semesters of technical school (see Table 4). For all types of training, higher means (more training) were apparent for the teachers who used microcomputers.

Pearson correlations were performed to investigate the relationship between the amount of training and attendance at computer planning sessions, both prior to computer purchase and for the school year studied. These correlations used the three most common sources of training: inservice hours, college credits, and semesters of technical school. All correlations were weak, but the correlation between inservice training and attendance at planning meetings during the current school year (.36) was more than twice as strong as the other variables measured.

Pearson correlations were also performed to investigate the relationship between the amount of training and favorable or unfavorable attitudes about computers (see Table 5). Two separate sets of variables measured the favorable and unfavorable attitudes. A Likert-type scale of 1–5 used to measure attitude indicated more intensely favorable and more intensely unfavorable attitudes, shown by higher numbers. The positive correlations between favorable attitude and the amount of training indicate that teachers with more training had more favorable attitudes toward computers. Negative correlations between the amount of training and unfavorable attitude indicate that teachers with the most unfavorable attitudes had the least amount of training.

Conclusions

Instructional designers can use the information revealed from this study to improve the state of instructional computing by directly responding to teachers' needs. Because teachers will mold any instructional design effort to their immediate classroom requirements (Shrock & Byrd, 1987), it is es-
TABLE 5
Correlation Table for Attitude and Training

<table>
<thead>
<tr>
<th>Correlations</th>
<th>R</th>
<th>R²</th>
<th>N</th>
<th>SIG*</th>
</tr>
</thead>
<tbody>
<tr>
<td>FAVORABLE ATTITUDE with:</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Inservice Hours</td>
<td>.1598</td>
<td>.0255</td>
<td>368</td>
<td>.002</td>
</tr>
<tr>
<td>College Credits</td>
<td>.1436</td>
<td>.0206</td>
<td>368</td>
<td>.006</td>
</tr>
<tr>
<td>Semesters Technical</td>
<td>.1279</td>
<td>.0163</td>
<td>368</td>
<td>.014</td>
</tr>
<tr>
<td>UNFAVORABLE ATTITUDE with:</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>College Credits</td>
<td>-.1128</td>
<td>.0127</td>
<td>368</td>
<td>.030</td>
</tr>
<tr>
<td>Semesters Technical</td>
<td>-.1093</td>
<td>.0119</td>
<td>368</td>
<td>.036</td>
</tr>
</tbody>
</table>

*One-tailed

It is essential that developers uncover the teachers' beliefs and incorporate such findings into design strategies.

Results of this survey highlight the importance of developing three distinct instructional computing efforts: computer-based instructional programs, user guides to accompany the programs, and teacher training for computer-based instruction. Although this conclusion is hardly surprising, instructional designers have yet to accept its emphasis on an integrated approach, as evinced by the fact that computer-based instructional programs are often produced without the crucial user guides and training to support them. Recommendations for instructional designers will center around these three areas of development.

In addition to matching the subject matter of the curriculum, developers should design programs which complement commonly used curricular materials and which fit the usage patterns imposed by equipment shortages, space limitations, and time constraints within schools. This type of program could accommodate the rationing of computer time among students and teachers as well as the restrictive patterns of a segmented school day by emphasizing either group activities at the computer or a combination of computing and non-computing activities. Such mixed learning activities might shorten the individual student's computing time by requiring that most of an assignment be completed without the computer.

Programs that offer fuller flexibility for curricular needs require some modification. Teachers want programs which include management capabilities and which allow simple modifications to provide a better curricular match. By developing programs that coincide with commonly used curricular materials, designers will facilitate on-site modifications that can help simplify and fine tune skills management. Further, designers should be aware that the increased popularity of "networkable" software stems from not only the ease of disk management but also from the enhanced capability for prescribing tasks and tracking student progress.

Development of User Guides

Teachers' demanding schedules and daily curricular requirements impose time constraints that limit the opportunity to locate, review, and plan for implementation of particular programs. User guides with good, complete documentation that follows a standard format would help relieve this situation. The guides should include a brief introductory section listing objectives, an overview of how the software works, implementation suggestions, and accompanying non-computer exercises that link the computer exercises to curricular topics. Although documentation should be complete, teachers need guides with a brief introductory section because they do not

...it is essential that developers uncover the teachers' beliefs and incorporate such findings into design strategies.
Successful implementation requires that teachers understand both the innovation and their role in using the innovation within the classroom setting.

Development of Teacher Training

Successful implementation requires that teachers understand both the innovation and their role in using the innovation within the classroom setting (Fullan, 1982). Therefore, teachers need adequate training, resources, and time for implementation (Weinschank et al., 1983). Instructional designers can help teachers by developing training that fosters meaningful computer implementation.

Teachers in this study were unsure about the objectives of instructional computing, and the majority named motivation as its primary benefit. Motivation precedes learning (Keller, 1987) but does not guarantee it. Teachers need training which addresses motivation and then goes beyond to the primary objectives and methods of instructional computing. Such training could suggest methods for linking computer activities to topics of study, for grouping students for cooperative work across ability levels, and for sharing resources among students so that all students gain meaningful, productive access to the machine. If all students were required to use the computer as a tool to complete significant class projects, then teachers could hardly continue the practice of requiring that "regular" class work be finished prior to using the machine.

Useful training also requires instructional designers to anticipate their audience. Generic training might be appropriate for preservice teachers, but meaningful inservice training must be tailored to the audience. Different training should be afforded to specific groups of teachers according to subject, grade level, and the teacher's skill level with computers. Further, because more experienced teachers might be more resistant to change (Cuban, 1986; Sarason, 1982), the age of the audience must be considered. Instructional designers must be aware of the possible reasons for resistance (e.g., job burnout, adherence to set routines, or status anxiety) and find a way to overcome them through training. Some training sessions might be appropriate for mixed audiences, but others may best address teachers' needs in separate groups. Whatever the case, staff development is one of the most influential forces affecting teachers' behavior (Shrock & Byrd, 1987). Instructional designers can influence teachers by informing them about the elements of an instructional design that implements computers in a strategic, systematic way that is meaningful to both students and teachers.

Training must be expanded into rural and underserved areas, perhaps through telecommunications or with the aid of the school library's media center (Schiffman, 1987). The media center could be responsible for organizing and administering training sessions, maintaining software collections, and consulting with teachers and students.

In addition to the obvious need for incorporating computer training requirements into preservice teacher training programs, school administrators must recognize the importance of inservice support for teachers during the implementation process. After all, the administrators are in the most influential position to help teachers (Weinschank et al., 1983). Implementation is an ongoing effort that goes well beyond the initial placement of computer hardware in the schools, so continuous training, instructional support, and good management are all vital to its success. Developers can work with administrators to suggest systematic strategies for providing ongoing support and incentives for teachers to attend inservice training. Components of such an approach could include plans for regular equipment maintenance and upgrades, software support, teacher training, and a method for evaluating implementation progress and adjusting the implementation plan accordingly.

In summary, instructional designers...
must realize that currently the implementation effort often ends with the placement of equipment in the schools, even though acquisition is just one of the first steps of successful implementation (Fullan, 1982). The same holds true for the products of instructional design—computer-based instructional programs, user guides, and teacher training all should be thoroughly tested prior to implementation, and most important, the effort to improve them should not simply end with their adoption by teachers or schools.

Computer-based instruction offers instructional designers a means of implementing instructional design principles, products, and procedures into the schools, but to take advantage of this opportunity, instructional designers must be aware of teachers' attitudes and concerns. To ensure successful implementation of instructional design products, developers must design instructional programs, support materials, user guides, and teacher training that are appropriate to the teachers' current needs.

The author gratefully acknowledges the insightful comments and suggestions offered by Dr. Norman Higgins during the development of this manuscript.

References


Knowledge Mapping: A Multipurpose Task Analysis Tool

Tim J. Esque
Intel Corporation
Chandler, AZ 85224

Abstract. A tool was developed to increase the objectivity and accuracy of task difficulty ratings for job design. The tool, knowledge mapping, involves identifying specific types of prerequisite knowledge for a given task and then assessing the difficulty of each type. The tool was applied in a semiconductor manufacturing environment and yielded valuable information not only for job design, but also for determining priorities for the development of training and establishing job levels for compensation purposes.

Much attention has been given to the myriad of task analysis tools and techniques available to educators and trainers. Some attention is given to integrating task analysis and instructional design (Carlisle, 1983; Reigeluth, 1983), but the majority of the efforts seem focused on distinguishing between the different task analysis techniques and providing some guidance as to how to choose the right technique for a specific application (Andrews & Goodson, 1980; Fosha, 1983; Jonsson & Hannum, 1986; Kennedy, Esque, & Novak, 1983).

One major distinction in determining how task analysis should be approached is whether the application is for industry or education. Kennedy, Esque, and Novak (1983), from a review of ten different methods, concluded that the most salient difference between approaches was whether the application was for industry or education. Jonsson and Hannum (1986) used the same distinction as the primary decision point in their algorithm for selecting the appropriate task analysis methodology for specific applications.

Differences in the desired outcomes of learning are often cited as the primary reason for differences between how task analysis is approached in industry and education. For example, due to its emphasis on cost effectiveness, industry is more interested in transferring procedural knowledge to quickly impact directly observable behaviors. On the other hand, education is more concerned with transferring nonprocedural knowledge for application to a broader range of situations over a broader time period (Jonsson & Hannum, 1986; Kennedy, Esque, & Novak, 1983). Desired outcomes of learning certainly impact one’s approach to task analysis; however, another major difference is that industry often uses task analysis to determine much more than the knowledge and skills required to perform a specific task.

Information from task analysis is applicable for several aspects of human resource management in industry, including job design, employee selection, and training those selected to do a job. The task information required for these applications often requires an approach to task analysis which is very...
The knowledge mapping technique was designed ... to merge three different classes of jobs ... into one class or family....

different from an approach directed solely at learning outcomes.

One major difference between different industrial approaches to task analysis is the kind of information collected about each task over and above how the task is performed and what knowledge and skills are required to perform it. Morsch and Archer (1967) list fourteen examples of what they call secondary rating factors in the task analysis process, including task criticality, task difficulty, satisfaction in performing the task, and supervision required while performing the task. Information from one or more of these task rating factors is typically used to make decisions about specific aspects of human resource management.

This article describes an approach to task analysis in which one of these secondary rating factors, task difficulty, was the primary source of information for making job design, selection, and training decisions in a semiconductor manufacturing setting. The key to this approach is a technique for deriving task difficulty information called knowledge mapping. Knowledge mapping was designed to improve the objectivity and the accuracy of task difficulty ratings for determining who is capable of performing what task and what training will be required. Although the technique was developed for a very specific application, it can be used to solve problems common to human resource management in any organization. Hence, it should be applicable in part or whole to many industrial settings.

The technique of knowledge mapping is described and examples of the tools that must be developed to implement it are provided. Lessons learned from the first application of the technique are cited, along with some discussion about the utility of the technique in general. First however, a description is given of the specific application for which this technique was developed. This should help any potential users of the technique to determine how and when it may be applied.

Throughout this article, many abstract terms such as context area, type of knowledge, and level of difficulty have been operationally defined in relation to each other for the purpose of describing the knowledge mapping process. The definitions are included (sometimes informally) to describe points that are useful in learning to apply the technique. They should not be interpreted as an attempt to structure the concepts of knowledge, but merely as a framework for understanding and using a practical tool.

Why Knowledge Mapping?

Although the decision to take a task-based approach to the job design process was made early on, the knowledge mapping tool emerged late in the planning process as a proposed solution to three distinct problems. The first problem was the scale of the effort. By far the most time consuming component of data collection would be prerequisite analysis, that is, identifying actual prerequisite knowledge and skills in objective form. With the analysis covering seven types of equipment, each requiring between one and two hundred discrete tasks, an exhaustive prerequisite analysis was out of the question. The problem was how to collect just enough information on each task to ensure that redundant training requirements across tasks would be easily recognizable so that tasks could be prioritized for further analysis.

The second problem had to do with making sound job design decisions. Job design is the process of making the decisions that determine (a) what tasks are performed by the workforce, (b) what tasks are to be clustered into what jobs, and (c) how the jobs are to be linked together (Davis & Wacker, 1982). An important objective of clustering tasks into jobs is that all the tasks in one job be of a similar level of difficulty. Otherwise, a choice must be made between selecting someone who is either under or over qualified for
The problem is that it is difficult to get unbiased data about the difficulty of a task. The traditional approach to collecting data on task difficulty (as well as most other secondary task rating factors) is relatively subjective. Typically, a generic 5- or 7-point scale is provided, and SMEs are asked to choose a single rating for each task (Morsh & Archer, 1967). Even when SMEs do not know how the data is to be used, they are inclined to inflate the importance and hence the difficulty of their job. This phenomenon becomes even worse when SMEs perceive a threat that their job is being designed out of existence, regardless of whether or not that is an objective of the analysis.

The third problem, like the first, concerns obtaining information about identifying and prioritizing training requirements, but also has implications for selecting candidates for a job. Although industry is more interested in procedural knowledge for changing directly observable behaviors, there are many tasks in an industrial setting that require the mastery of non-procedural knowledge. In Gagné's terminology, procedural knowledge would be the application of discrimination, concrete concepts, and rules to learning the step-by-step instructions for performing a procedure. Nonprocedural knowledge, on the other hand, includes the remaining intellectual skills (defined concepts and higher order rules) and would typically require the use of cognitive strategies for applying knowledge to making decisions, developing motor skills, or solving unique problems. Gagné defines each of these intellectual skills in The Conditions of Learning (1977). Their use is applied here to provide an operational distinction between the terms procedural and non-procedural knowledge, not necessarily in the context of Gagné's taxonomy.

The distinction as to whether a task requires procedural knowledge, non-procedural knowledge, or both provides a framework for determining the type and amount of knowledge and skills required to perform the task. This information is the basis for determining whether or not to train a given population or to select a population with the appropriate educational background. It also makes it possible to identify standardized instructional approaches for each anticipated type of training, which means training development can be prioritized and parcelled out early in the process without the risk of designing incompatible materials or developing them for the wrong audience. The problem, then, is how to determine the basic types and amount of knowledge and skills required to perform specific tasks even before the content of the training—specific prerequisite knowledge and skills—is known.

**The Knowledge Mapping Technique**

Knowledge mapping involves identifying for a given task specific types of prerequisite knowledge and cognitive skills within defined content areas, and then assessing the difficulty of each type of knowledge and skill. The only difference between a type of knowledge or skill and a content area is level of specificity. Content areas represent a very broad subject matter; types of knowledge and skills represent a more specific subject matter within a content area. The defined content areas must encompass all expected types of knowledge for the specific application. Figure 1 shows the nine content areas used to analyze semiconductor manufacturing tasks. Ideally, all types of knowledge required to operate, maintain, or monitor the processing of semiconductor manufacturing equipment could be categorized within one of these nine content areas.

Three different levels of task difficulty were defined based on their usefulness in determining who can be effectively trained for what job and what general form of training (from a limited number of selected forms) can be applied to the task. For this reason, the levels do not, and were not intended to, fit precisely into the existing task, knowledge, or learning taxonomies (e.g., Bloom et al., 1956; Gagné, 1972, 1977; Reigeluth, 1979). Figure 2 shows the generic levels of task difficulty.

The definitions are considered generic because they are modified in actual use for each specific content area. The modification is intended to help the SME make the transition from distinguishing between levels of difficulty across tasks to distinguishing between levels of difficulty for types of knowledge required to perform a task. Because of this, the specific definitions within a content area typically do not include the goal of the task (follow step-by-step directions, perform motor

---

**Figure 1. Content Areas for Semiconductor Manufacturing**

- Technologies/Disciplines
- Equipment Assemblies
- Generic Process Knowledge
- Specific Process Knowledge
- Quantitative Skills
- Tools/Materials Knowledge
- Computer Skills
- Safety Knowledge
- Problem Solving Skills

---

**Figure 2. Generic Levels of Task Difficulty**

**Level A** Tasks that require the performer to make discriminations and/or recall only concrete concepts and simple rules to understand and follow step-by-step directions.

**Level B** Tasks that require the performer to use limited knowledge of a related set of defined concepts and principles from one or more technologies (e.g., electronics, mechanics, optics, gas dynamics, etc.) to correctly perform motor skills or to make decisions.

**Level C** Tasks that require the performer to understand the underlying concepts and principles from one or more technologies and have the ability to apply that knowledge to solve unique problems.
skills, make decisions, solve unique problems) as stated in the generic definitions. Figure 3 shows how the generic levels of task difficulty have been modified for three of the nine content areas for semiconductor manufacturing. These specific content area definitions are referred to as knowledge-level definitions.

Figure 4 shows the knowledge mapping information for one sample task. This same information is collected for each discrete task in the job being analyzed. Note that the nine numbered subsections are the determined content areas shown in Figure 1. Although these content areas are broad, they are application specific.

Listed below each content area are the types of knowledge within that content area required to perform the task as solicited from the SMEs. (Areas 5, 7, and 8 differ in this respect because the nature of these content areas does not require listing types of knowledge.) Each of these types of knowledge have then been rated by the SMEs as A, B, or C, based on the level of difficulty of the type of knowledge required to perform the task. The ratings A, B, and C correspond to the knowledge level definitions (see Figure 3). The highest difficulty rating for any one type of knowledge required for a task is the difficulty rating for that task (difficulty ranking is ascending from A to C). The task in Figure 4 would be designated as a B-level difficulty task. The end result of the knowledge mapping process is a matrix showing the difficulty level of each task, broken down by content area and type of knowledge (see Figure 5).

Using the Knowledge Mapping Data Base

Together with frequency and duration data for each task, the knowledge mapping data provides enough information for the SMEs and management to make informed job design decisions. The task difficulty ratings for each task provide information about who is and who is not qualified to perform that task. They also provide information about the amount of training required over and above a person's existing knowledge. For example, one can conclude that a task of A-level complexity could be taught with a minimal amount of instruction to almost anyone who has at least a secondary education. Tasks of B-level complexity can also be taught to almost anyone with a secondary education; however, the amount of training required to learn the task will vary with each individual's background. Each individual may have a different amount of knowledge of the specific set of related concepts and principles needed to perform the task. (This is a given for this particular application because worker populations already exist for each of the three separate functions.) Although this information alone does not prescribe who is capable of performing a task or how much training any one individual would need, it provides a framework for getting that information through interviews (to record past experience) or through testing.

Tasks of C-level complexity provide more straightforward information for the job design process. A task of C-level complexity can only be taught to a person with specific postsecondary education or an equivalent amount of on-the-job experience. In other words, because the task requires knowledge of underlying principles in one or more disciplines, the job requirements typically can be defined in terms of educational background.

All of this information is useful for designing new jobs that are cost-effective in that they are made up of tasks
**Figure 4. Completed Data Collection Sheet**

**TASK:** Check Overlay

<table>
<thead>
<tr>
<th>1. Equipment Assemblies</th>
<th>A</th>
<th>B</th>
<th>C</th>
</tr>
</thead>
<tbody>
<tr>
<td>Front Control Panel</td>
<td>✓</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Mask Transport System</td>
<td>✓</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Wafer Handling System</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>2. Technology/Discipline</th>
<th>A</th>
<th>B</th>
<th>C</th>
</tr>
</thead>
<tbody>
<tr>
<td>Optics</td>
<td>✓</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Pneumatics</td>
<td>✓</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Mechanics</td>
<td>✓</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>3. Tools/Materials Knowledge</th>
<th>A</th>
<th>B</th>
<th>C</th>
</tr>
</thead>
<tbody>
<tr>
<td>Overlay Test Set</td>
<td>✓</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Microscope</td>
<td>✓</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>4. Generic Process Knowledge (Process Parameters)</th>
<th>A</th>
<th>B</th>
<th>C</th>
</tr>
</thead>
<tbody>
<tr>
<td>Critical Dimensions</td>
<td></td>
<td>✓</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>5. Specific Process Knowledge: (circle one)</th>
<th>A</th>
<th>B</th>
<th>C</th>
<th>N/A</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>6. Safety Knowledge:</th>
<th>A</th>
<th>B</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>UV Light Exposure</td>
<td>✓</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>7. Computer Skills: (circle one)</th>
<th>A</th>
<th>B</th>
<th>C</th>
<th>N/A</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>8. Troubleshooting Skills: (circle one)</th>
<th>A</th>
<th>B</th>
<th>C</th>
<th>N/A</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>9. Quantitative Skills:</th>
<th>A</th>
<th>B</th>
<th>C</th>
</tr>
</thead>
<tbody>
<tr>
<td>Solve Simple Equations</td>
<td>✓</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Figure 5. Partial Knowledge Mapping Data Base for a Diffusion Furnace

<table>
<thead>
<tr>
<th>CONTENT AREAS:</th>
<th>EQUIPMENT ASSEMBLY</th>
<th>TOOLS/MATERIALS</th>
<th>TECHNOLOGIES/DISCIPLINES</th>
<th>ETC.</th>
</tr>
</thead>
<tbody>
<tr>
<td>TYPES OF KNOWLEDGE:</td>
<td>COLD START UP</td>
<td>CLEAN LUBE BOATLOADER</td>
<td>VERIFY MFC FAILURE</td>
<td>CHANGE QUARTZ TUBE</td>
</tr>
<tr>
<td>TASKS</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

with similar knowledge requirements. In addition, the same information can be used to estimate how much training specific populations or individuals require in order to perform the job. This is valuable for determining which existing population is best suited to learn and perform the newly designed job(s).

Why take the time and effort to ensure that task complexity ratings are tied to a common set of definitions? Job design and selection decisions are only as good as the data from which they were derived. Defining difficulty at a subtask level (i.e., each type of knowledge within each content area) and then converting it to a single rating for each task has several advantages over rating difficulty just at the task level. The knowledge mapping technique forces the SME to think about all aspects of the job tasks, and hence reduces the probability that any prerequisites to performing the tasks will fail to be considered. One type of prerequisite knowledge could be the difference between whether a task is of A, B, or C level of difficulty. Even more important, rating difficulty at a subtask level makes it much easier for the experienced interviewer to probe the SMEs on their responses, even when the interviewer is relatively naïve about how the task is performed. This process is demonstrated later in this article.

Once the job design has been completed and the target population has been selected, knowledge mapping information provides a structure for establishing training development priorities before an in-depth prerequisite analysis has been conducted. The first step of establishing training development priorities is to sequence the tasks for training. The knowledge mapping data, which lists the types of required knowledge for each task along with an understanding of when each task is performed, provides a basis for clustering tasks by how and when they will be trained. Once a sequence of training has been developed, the order in which the identified units of training will be developed needs to be prioritized based upon the sequence of training, what training already exists, and what skills and knowledge the target population for the specific job already possesses.

After determining the order in which training will be developed, the knowledge mapping data can be used to parcel out development responsibilities to several parties. Tasks that require similar areas of knowledge can be grouped for development by the same individuals or teams so that training for the same knowledge and skills is not developed twice. For example, the knowledge map for semiconductor manufacturing jobs reveals several tasks requiring prerequisite knowledge of basic photolithography principles. Having identified this before beginning training development, a single effort can be mounted to develop or acquire instruction on basic photolithography designed specifically to prepare the learner to perform a certain set of tasks. In addition, the generic difficulty levels can be used to categorize types of training development so that all development going on in parallel will yield similar products for similar applications. For example, methods and formats for developing procedural versus non-procedural training can be determined before training development begins. The A, B, and C knowledge level definitions from the knowledge mapping data can be used to identify whether knowledge is procedural or non-procedural and to prescribe the method and format of the training materials. This standardization strategy can also
be used for involving SMEs without training development skills in the process. Once the methods and format for training have been determined, tools for soliciting from SMEs the right information in the right form can greatly increase the efficiency of a large-scale development effort.

A final use of the knowledge mapping, unanticipated in the initial application, is establishing the comparable worth of the newly designed jobs. The task difficulty ratings provide a quantifiable measure of job difficulty that can be used for equalizing compensation with other jobs, as long as the other jobs have also been knowledge mapped. For example, a job made up of 20% A-level and 80% B-level tasks can be quantified as more difficult than a job made up of 80% A-level and 20% B-level tasks. Job difficulty is, of course, not the only factor in assessing the comparable worth of jobs. However, when difficulty is defined based upon the prerequisite education and training required, it is a major determinant. This was especially valuable in the initial application where new, enhanced jobs were derived from three existing jobs, all of which were documented through knowledge mapping.

Developing Knowledge Level Definitions

Before knowledge mapping data can be collected, the primary tool for knowledge mapping must be developed: knowledge level definitions. There are two major steps to developing knowledge level definitions for the analysis of specific tasks or jobs. The first step is to identify discrete content areas appropriate to the tasks or jobs. Once these have been defined, the second step is to translate the generic levels of task difficulty (see Figure 2) into three area specific definitions for each defined content area (see Figure 3). The area-specific knowledge level definitions should reflect a hierarchy of difficulty parallel to the generic levels of task difficulty. Each step is described below.

Identifying Content Areas

Knowledge mapping requires identifying discrete content areas associated with the job(s) being analyzed. Different jobs require knowledge of different content areas. Content areas can be defined at various levels of specificity. For example, knowledge of physics is a content area, so is knowledge of photolithography, and so is knowledge of the semiconductor applications of photolithography. Of course, for knowledge mapping to be a practical tool, content areas need to be defined at a similar level of specificity. The level of specificity also needs to be a relatively global one. Otherwise, the length of time to analyze any one task would be prohibitive.

In order to attain an appropriate level of specificity, global content areas must first be identified for the job(s). These global areas then need to be broken into more specific areas until the level is reached where it is easy to list specific types of knowledge within the defined content areas. A type of knowledge would be a more specific subject matter within a content area. Within the content area of quantitative skills types of knowledge would include knowledge of simple math, algebra, and statistics. The process is best understood by looking at an example.

Figure 6 shows the breakdown of semiconductor manufacturing knowledge into three increasingly more specific levels of content areas. The first level represents the two major content areas that make up knowledge of semiconductor manufacturing. Semi-
conductor equipment refers to all the knowledge and skills required to operate and properly maintain and troubleshoot semiconductor manufacturing equipment. Semiconductor Processing refers to all the knowledge and skills required to characterize, monitor, maintain, and improve a semiconductor manufacturing process. Level 2 breaks down the two major content areas to the next level of specificity. The third level is only partially broken out because the number of discrete content areas increases greatly from level 2 to level 3. In addition to the increase in the number of items, it is possible to look only at the items in level 3 and accurately conclude that they are types of Technologies/Disciplines. In contrast, it would not be accurate to conclude that the items breaking out of Semiconductor Equipment in level 2 are types of semiconductor equipment. These are both signs that the previous, most general level (level 2 in this case) is the appropriate level to use in developing knowledge level definitions.

It is important to note that within each level of specificity, the listed content areas are discrete or mutually exclusive of each other. If two content areas are found to overlap, either the areas need to be redefined or possibly a third area needs to be created. Significant overlaps between the content areas can result in the collection of redundant data.

The process of defining content areas requires SME assistance unless the person developing the definitions is familiar with the job(s). At the very least, SMEs should check the defined content areas after a first draft has been developed. If the number of content areas exceeds ten, they are probably at too specific a level or the job(s) being analyzed may not be practically analyzed through knowledge mapping.

Developing Knowledge Level Definitions for Each Content Area

This is by far the most difficult aspect of preparing to conduct knowledge mapping. There were no set procedures identified for accomplishing this step during the initial application of knowledge mapping; however, a general description of how the definitions were derived will provide a starting point for future users of the knowledge mapping technique.

The difficulty in achieving good knowledge level definitions lies in the fact that developing the definitions requires a thorough understanding of both the generic levels of task difficulty and the specific subject matter. Because the individuals with the deepest understanding of the generic levels of task difficulty (instructional designers) are typically a different population from those who have the deepest understanding of the specific subject matter (SMEs), the challenge is having individuals from each of these populations work together to achieve knowledge level definitions with which they are both comfortable.

In the initial application, the process began by identifying individuals with knowledge of instructional design and experience in applying that knowledge to the given content areas (i.e., equipment assemblies, process knowledge, computer skills, etc.). When these individuals had developed first-draft knowledge level definitions, they were reviewed by SMEs. After the SMEs gave their suggestions and all involved parties were comfortable with the revised definitions, the definitions were tried out with a sample of SMEs from the population that would be used in the actual knowledge mapping interviews.

Like specifying the content areas, the development of knowledge level definitions is an iterative process. Changes to the initial agreed-upon definitions were made through several tryouts of the definitions until the decision was made to "freeze" the definitions for the initial application and record any potential improvements for use in future applications. One general strategy for improvement that came up several times during the first application and will no doubt be used in future applications involves providing specific examples for each knowledge level definition. This strategy is discussed later.

Collecting Knowledge Mapping Data

The knowledge level definitions are used in an interview setting after a list of job tasks has been developed. Carlisle (1986), Kennedy, et al. (1983), Andrews and Goodson (1980), and many others provide descriptions and bibliographies of documented techniques for generating task lists. Whatever techniques are used to generate the preliminary task list, it should be checked by SMEs before beginning the knowledge mapping interviews. The objective of a knowledge mapping data-collection interview is to identify for each task the types of knowledge required to perform the task and the level of difficulty of each. However, before the interview can be conducted, the interviewer needs to do some preparation beyond development of the definitions.

Preparing for the Interview

The interviewer needs to take three things into the knowledge map data collection interview: a copy of the knowledge level definitions, a data collection sheet for each task, and a predetermined list of types of knowledge for certain content areas. The definitions should be in a form useful to the SMEs who will be interviewed, and should be clearly organized (by content area) and concisely written. The data collection sheet should list each content area along with space to write in or check off specific types of knowledge. Adjacent to each type of knowledge should be a space to designate the level of difficulty (A, B, or C; see Figure 4). Finally, to ensure that types of knowledge are kept at a consistent level of specificity across interviews, a list of likely types of knowledge should be generated for the appropriate content areas. This can be done in the initial interviews for generating task lists by explaining to the SMEs what "types of knowledge" are, and then having them generate as exhaustive a list as they can for the given content area. For example, referring back to Figure 6, an experienced semiconductor operator should be able to list types of equipment assemblies (e.g., control panel, wafer handling assembly, mask transport system) at a consistent level of specificity. Other SMEs may generate new types of knowledge within the same content area during the knowledge map data collection, but they are more likely to stay at the same level of specificity if examples are provided.

With all of the tools in place, the interviewer can further prepare by getting a general idea of the job(s) they intend to analyze. This can usually be achieved adequately by informally ob-
serving someone performing the job(s) and asking a few questions. The goal is to pick up enough terminology to talk about the job intelligently and also be able to visualize the SME's description of how certain tasks are generally accomplished.

Conducting the Interview

The SME should first be briefed on the purpose of the interview and how it will be conducted. Then, for each content area (for example, Power Systems Components or Quantitative Skills from Figure 6), the SME should be asked if that content area is in any way required for performing the specific task. If the answer for that content area is yes, the next question should be “what types of knowledge within the content area are required to perform this specific task?” The SME should be shown the predetermined list of types of knowledge to make sure the appropriate level of specificity is adhered to. However, responses should not be limited to the predetermined list.

When all appropriate types of knowledge have been listed, the SME should be asked, for each specific type of knowledge, “what level of this type of knowledge is required to do this specific task?” The SME should use the definitions to determine the level of knowledge required. To ensure that the SME is adhering to the definitions, the interviewer should probe to make sure the correct level has been identified. Figure 7 provides an example dialogue between the interviewer and the SME for one type of knowledge within one content area for one task. Note how the interviewer probes to verify that the SME is sticking to the definitions. The probing is only necessary on a periodic basis after the interviewer knows that the definitions are being adhered to consistently.

Lessons Learned

In its first application, knowledge mapping has shown its value by adding objectivity to SME ratings of task complexity, creating a time saving structure for training development and delivery, and providing a quantitative basis for compensation analysis. It is anticipated that the tool can be used

Figure 7. Excerpt from a Knowledge Mapping Data Collection Interview

Interviewer: The next task is cleaning the etcher chamber. Using our provided listing, what etcher assemblies do you need knowledge of to clean the etcher chamber?

Interviewer: Oh let's see... you would need knowledge about the interlock mechanism and quite a bit of knowledge about the etcher chamber itself.

Interviewer: Think about performing the task. Are there any others that we’ve left off the list?

Interviewer: Well, you need to know about O-rings, but I guess that would be considered a part of the chamber which we already listed.

Interviewer: OK, now for the chamber. What level of knowledge would you need to have about the chamber to safely and consistently perform this task? Look at the knowledge definitions under “Equipment Assemblies.”

Interviewer: (Looks at definitions.) I'd say level B.

Interviewer: What technologies or disciplines do you need to understand about the chamber to clean it effectively and safely?

Interviewer: Well, basically you need to understand how the chamber physically works, which requires knowledge of several basic principles of vacuum technology. Without this knowledge, chances are you will cause the chamber to leak when it's back in operation.

Interviewer: Could the knowledge required be documented in the form of a series of procedures?

Interviewer: Not really. You tend to have to make decisions that require background knowledge.

Interviewer: OK, that certainly falls within level B. Now, what level of knowledge do you need about the interlock mechanism to clean the etcher chamber effectively and safely?

Interviewer: Oh, I'd say level C on that one.

Interviewer: OK, why is it that you need such in-depth knowledge of the interlock mechanism?

Interviewer: Because if you fail to seal the interlock correctly, you could have a potentially dangerous mix of gases during operation.

Interviewer: You definitely don't want to do that, but unless you need in-depth knowledge of the interlock mechanism to avoid making a mistake, we probably need to address the danger in the “Safety Knowledge” section.

Interviewer: I guess that's true. You don't really need to know much more than a couple of hydraulics principles to successfully perform this specific task, so I guess that makes it level B.

Interviewer: I agree. OK, if that's all the etcher assemblies involved in this task, let's move on to the next content area.
more efficiently in the future based on lessons learned from this initial application.

Developing the Definitions

One comment which was repeated several times by SMEs participating in knowledge mapping interviews was that the definitions would be easier to understand if specific examples were given for each definition in each content area. This advice was followed in a second application of knowledge mapping that focused on nonroutine tasks that fall into the same three semiconductor manufacturing functions as the initial application (operations, equipment, and process). Figure 8 shows the examples provided for the different definitions within one content area.

Although each is based on one specific piece of equipment, the examples typically focus on subject matter and terminology that is common to many different kinds of equipment. Since the inclusion of these examples, many participating SMEs have commented that the examples do indeed help distinguish between levels. The participating analysts also are in agreement that the examples make it easier to explain to the SMEs the differences between the generic levels of task difficulty: A, B, and C.

Before Beginning the Interview Process

Once a set of clearly and concisely written knowledge level definitions with well thought out examples is created, the key to using the definitions effectively is for all users of the definitions (analysts) to interpret them consistently. The best way to achieve this goal is to limit the users of the tool to as small a group as possible, preferably the same group that developed the definitions. This will not always be practical, especially for larger scale applications. The alternative is to provide training for the users. Several observations about the preparatory training were made during the initial application.

One mistake to avoid is combining the knowledge mapping training with other information and training. In the initial application, travel cost was the impetus for packing too much training into one day. The savings, however, ended up being used to evaluate and coach the users later at their respective sites. The ideal objective of the training would be to achieve a significant interrater reliability. However, even if this goal is not attempted (as it was not in the initial application), plenty of time needs to be allowed to go over the definitions and ensure that everyone interprets them the same. Once this has been achieved, it is critical to provide practice and coaching on conducting the interview. If a realistic simulation of a knowledge mapping interview cannot be carried out in the classroom, each interviewer should be supervised and coached during their first interview.

One positive interviewer attribute, which is very difficult to train, is recognizing when the SME is no longer attending to the interview questions. This is often a risk with SMEs who catch on quickly and do not seem to need reminders about accurately using the definitions. Unless each consecutive task is very different in nature, it is easy for fast learners to start answering the questions in patterns based on key verbs in the task statement. It is important that the interviewer recognize when this is happening and do something to force the SME out of patterned responses. It often helps to try to mix up the tasks so that you do not analyze too many similar ones in a row.

Another factor for successful application of knowledge mapping is gaining access to the appropriate SMEs. It is very important that the SME used to provide knowledge map data on a group of tasks be very knowledgeable on how those tasks should be performed. This person should also be very knowledgeable about what someone needs to know to perform the task correctly, which is what the knowledge mapping tools are designed to find out.

The most appropriate SME for the knowledge map data collection interview may well be someone other than the SMEs used to derive the task list. Although the people who actually perform the job can best describe what tasks they perform, in many cases, they are not necessarily the most knowledgeable about why tasks are done in a specific way. As a general example, where technicians are assisting en-

![Figure 8. Knowledge Level Definitions and Specific Examples.](image)

**Equipment Assemblies**

A. State the function and location of the assembly and its key parts. Describe the general sequence of events for how the assembly/part works.

*Example:* To clean the boatloader, you need to have the following knowledges about the boatloader assembly: state key parts and where they are located, and describe the sequence of how the boatloader operates because you need to partially operate the boatloader while you clean it.

B. Use limited knowledge of one or more technologies (e.g., electronics, mechanics, pneumatics, optics, vacuum, etc.) to describe why this specific assembly functions the way it does in a given situation.

*Example:* To adjust paddle soft placement speed, you need to have the following knowledges about the paddle: Explain how the vacuum system makes the paddle move in order to determine what kind of vacuum system adjustment would cause the paddle to sit further forward or backward.

C. Use broad-based knowledge of one or more technologies to describe why this general type of assembly generally functions the way it does.

*Example:* To calibrate any MFC in the Fab, you need the following knowledge about the MFC: broad-based knowledge of electronics and gas flow dynamics in order to explain how MFCs become out of calibration.
...it is important to identify the one SME who is most knowledgeable about how the tasks should be performed for each job....

giners in an industrial setting, the technicians can probably provide a more accurate description of their job, but the engineers probably have more knowledge about why specific tasks are done in specific ways.

It is a good idea to knowledge map the first few tasks with two or three different SMEs and resolve any discrepancies that occur due to the knowledge level definitions or interviewer interpretation. When an acceptable level of cross-validation is apparent, it will be adequate to knowledge map the rest of the tasks in the same area of expertise with one SME. Hence, in setting up an SME base, it is important to identify the one SME who is most knowledgeable about how the tasks should be performed for each job or area of expertise.

The resulting data base is only as useful as the SME is knowledgeable. The most knowledgeable SMEs are also the most valuable on the job (assuming an industrial application), so it is often difficult to get management to release them. The alternative, however, is to risk either ending up with an inaccurate data base or performing time-consuming interviews on the same tasks more than once. Even if getting the most knowledgeable SMEs extends the project timeline, it is worth the wait.

Successfully Conducting Knowledge Mapping Interviews

Like any other interview situation, there is a limit to how long the interview can be conducted productively before either the analyst, the SME, or both become fatigued. Because the analysis of each type of knowledge can potentially alter the difficulty level for the whole task, it is critical that the interview be terminated before either party becomes fatigued and therefore careless.

In the initial application, all interviews were scheduled for a two-hour duration, allowing for 20 minutes of explanation, 90 minutes of productive knowledge mapping, and 10 minutes to spare. It turned out that productive knowledge mapping could be sustained for no more than 60–75 minutes. For this reason, the ideal duration for an entire knowledge mapping interview seems to be about 90 minutes.

Finally, the most common mistake made by interviewers was to forget to specify early in the interview the conditions and criteria for performing the task. In other words, instead of asking, "what level of this type of knowledge do you need to perform the task under ideal conditions (no system failures) without risk to yourself, the product, or the equipment?" the interviewer would ask "what level of this type of knowledge do you need to perform the task?" Without the specific conditions and criteria, the SMEs are likely to overestimate the knowledge required because they often have more knowledge than is required to perform the task under ideal conditions. Conditions and criteria for performing the task could be different for separate applications, but it must be determined initially and communicated consistently to participating SMEs.

Discussion

Several benefits of using the knowledge mapping technique to accomplish various aspects of human resource management—job design, training curriculum design, and establishing job levels for compensation—have been presented. However, conclusions about the global value of knowledge mapping cannot be drawn from observations of the initial application of the technique. This is because, first, the technique was designed for and around this initial application, and second, no systematic, data-based assessment of the intervention has been conducted. At this point, the best indication of the tool's utility is simply that management has chosen to expand the use of the tool beyond the initial tryout described in this application. In the absence of data-based conclusions about the technique, some hypothetical pros and cons of knowledge mapping are presented.

...knowledge difficulty seems to be an adequate indicator of who is capable of doing what.
There were two primary reservations about the knowledge mapping tool which were discussed at some length both before and during the initial application. The primary problem, as perceived by many people associated with the project, was that the cost in SME and analyst time of developing the tools and collecting the data would outweigh the anticipated benefits of the technique. The second major reservation about the technique, prior to its use, was whether a measure of task difficulty based solely on knowledge and cognitive skills, without incorporating motor skills, would be adequate for making critical job decision.

The costs versus benefits concern was a demonstration of a classic conflict in for-profit organizational settings: how to balance up-front data collection and analysis against trial-and-error, “experienced-based” decision making. More specifically, there was a strong contention that a team of expert performers representing each of the three job functions being analyzed should be put into the work environment with the objective of determining how to optimize the efficiency of human resources in sustaining the specified operations. Proponents of the knowledge mapping technique maintained that the real cost of this alternative method could very likely be as great or greater than the cost of knowledge mapping, depending, of course, on the abilities of the individuals on any one team. It was finally decided to use knowledge mapping in the initial set of applications because the alternative method would not yield the level of documentation required to develop training. Again, the fact that the use of knowledge mapping has extended beyond the initial application is the only indication that the technique is cost effective.

The second reservation, whether task complexity could be adequately measured based solely on knowledge rather than observable skills, was not openly discussed but was very much a concern of the designer of the technique. After all, the vast majority of the tasks in question manifested at least partially in the form of observable behaviors. The decision to focus strictly on knowledge and cognitive skills in assessing task complexity was based on two assumptions. First, the assessment of skills in addition to the knowledge required to perform a task significantly increases the cost of data collection and analysis. Second, assessment of skills is much more subjective than assessment of knowledge, and would therefore require a technique much more rigid than interviewing.

One informal observation of the result of the knowledge mapping technique is that knowledge difficulty seems to be an adequate indicator of who is capable of doing what. There were very few cases where a task designated as appropriate for a specific employee population was found in practice to be inappropriate due to the difficulty of the motor skills required. (Some tasks were reallocated due to safety considerations, even though safety was included as one of the nine major areas of knowledge.) One interpretation is that even though a job is behaviorally oriented, the actual limits to performance often are not knowing what to do and when to do it. Based on this line of reasoning, it would be expected that in a situation where tasks are clearly dependent on motor skills, the knowledge mapping technique is not likely to be as useful.

In summary, knowledge mapping was designed to improve the objectivity and accuracy of task difficulty ratings for determining who is capable of performing what tasks and what training is required. In this particular application, the contribution made by knowledge mapping to job design, training curriculum development, and determining the relative worth of tasks and jobs warranted future use of the technique for similar applications. However, having been developed in an industrial setting, the methodology was applied at a very formative stage and without quantitative evaluation. It is hoped that by sharing our experience in this article, the methodology will be further analyzed and developed both as a practical tool and in the context of existing models of learning and performance.

---

Knowledge Mapping was developed with the help of many Intel colleagues, in particular Anne Polino and Paul Valle. Thanks also to the JID consulting editors who commented on earlier drafts of the manuscript for providing helpful and insightful analysis of the technique and its description.

References


Elena D. Beattie  
ERIC Clearinghouse on Information Resources  
Syracuse University


The programs described in this sourcebook of instructional and learning support activities in Canada that use communications technology in distance learning is based on the findings of a survey of Canadian universities and colleges, provincial departments of education, the telecommunications industry and organizations, professional associations, and private corporations. Included are both public and private sources that directly support the instructional process as well as those that support the administration of distance education projects and research. The first of three sections of this report provides background information on the survey and an introduction to the sourcebook. The second section comprises three charts which summarize the communications technologies used by each of the respondents, the educational levels for which their instructional programs are offered, and the subject areas covered by these programs. The third section, which makes up the major part of the report, presents detailed descriptions of 57 projects, summarizes satellite and Telidon field trials or experimental projects, and identifies some additional projects using communication technologies in instruction as well as in research and electronic messaging. A glossary of terms is provided, and three appendices include a description of the survey methodology, the survey questionnaire, and a feedback form for the readers of this sourcebook. —Microfiche 82t; paper copy $21.34, plus shipping, as document ED 290 464.


This booklet begins by examining the role played by evaluation in course and curriculum development, and then shows how the basic “error elimination” approach advocated by the philosopher Karl Popper can be used as a basis for the ongoing evaluation of instructional systems. Next, two contrasting paradigms of evaluation are described, one that concentrates on measuring the outcomes of the instructional system (the agricultural/botanical or scientific approach) and one that pays more attention to what happens during the educational process itself (the social/anthropological or illuminative approach). Finally, five diagnostic techniques commonly used as part of an evaluation strategy are reviewed and the respective uses, strengths, and weaknesses of each are discussed: (1) results from student assessment; (2) student questionnaires and interviews; (3) observation of the instructional system in progress; (4) feedback from teaching staff directly involved with the instructional system; and (5) feedback from people having an indirect link with the instructional system. Models of the systems approach to course development, the role of an instructional system, and the general methodological approach advocated by Popper are provided, as well as extracts from a Likert scale and a semantic differential scale taken from course evaluation questionnaires. An annotated list of three items recommended for further reading is included. —Microfiche 82t; paper copy $1.94, plus shipping, as document ED 289 507.


This booklet is the first of four sequels to the guide “Educational Objectives” that discuss the selection and use of instructional methods. Following a brief introduction, the systems approach to course or curriculum design is reviewed, and the selection of
appropriate instructional methods is described as the second key step in the course or curriculum development process. In addition, the wide range of instructional methods available to teachers and lecturers are discussed, and three broad categories of methods are identified—mass instruction techniques, individualized learning techniques, and group learning techniques. Finally, the range of techniques available within each category is examined, and the educational strengths and weaknesses of each technique are discussed. The text is supplemented with two figures and four tables, and two sources for further reading are listed.—Microfiche 82; paper copy $1.94, plus shipping, as document ED 289 485.


This paper introduces the concept of radio-assisted practice (RAP) and outlines the nature and initial findings of a British research project which is investigating the potential of RAP in preservice teacher education. The paper falls into three main parts. The first situates matters in terms of information-processing skill (IPS) psychology and comments on its recent application to thinking about teacher expertise. It is also noted that recent work on professional competence, as well as student-teacher construals, corroborates the IPS implication that the development of teaching expertise requires meaningful classroom practice. Presenting RAP as an unobtrusive means of guidance using miniaturized radio communication during ongoing teaching, the second part of the paper distinguishes the technique from the concept and offers some principles and corresponding rules of thumb for “tapping.” The third part describes and presents some preliminary results of an 18-month research project designed to assess the efficacy of RAP for student teachers and ways of initiating supervisors into RAP usage. A 17-item reference list is provided.—Microfiche 82; paper copy $3.88, plus shipping, as document ED 288 492.


This study examined the relative effectiveness of different media in presenting modeling stimuli to adult trainees in a supervisory skills training program and assessed their satisfaction with the training and their performance. Eighty undergraduate students were randomly assigned to one of four conditions—audio only, text only, audio plus text, or video plus audio—for training which focused on developing their skills in coaching an employee for improved performance. A videotape with or without the video image and a typewritten transcript of the videotape were used in different combinations to present the training, which consisted of an introductory discussion of coaching by a trainer, a modeling example in which a supervisor coached an employee using six key behaviors, and a group discussion in which the trainer and a group of trainees (actors) discussed and critiqued the model's use of the key behaviors. It was found that all four conditions were equally effective in terms of learning recall, although audio alone was found to be superior to video plus audio in terms of actual use of the newly acquired supervisory skill. Nonverbal behavior was not affected by the media, nor were there any significant differences in training satisfaction as a function of the media used. A 27-item reference list is provided.—Microfiche 82; paper copy $1.94, plus shipping, as document ED 287 464.

The above documents may be ordered from the ERIC Document Reproduction Service (EDRS). 3900 Wheeler Ave., Alexandria, VA 22304. $1.00. Please order by ED number, indicate the format desired (microfiche or paper copy), and include payment for the price listed plus shipping. For VISA or MasterCard orders or information on shipping costs, call EDRS at 1-900-227-3742. Inquiries about ERIC may be addressed to the ERIC Clearinghouse on Information Resources, 830 Huntington Hall, Syracuse University, Syracuse, NY 13244-2340, (315) 443-3640.