Designing Technology-Based Performance Support

Kent L. Gustafson
Contributing Editor

> Instead of struggling to train your corporate staff to use your mission-critical applications, why not have the application help teach the users to perform their jobs?
>
> (Foster, 1997)

This quote from Ed Foster nicely captures the heart of the issue of on-the job performance support versus training. Ultimately, organizations seek meaningful performance from employees, with training being a means, not an end. Increasingly, these organizations are turning to tools and systems that support workers while they are doing real work to overcome the known limitations of training, including lack of transfer, forgetting, and rapidly changing workplace conditions. Performance support, particularly in technology-rich work environments, shows considerable promise for enhancing performance while reducing training time and costs. This trend has major implications for professionals engaged in conventional forms of training development.

Below I discuss several questions commonly asked about technology-based performance support. I have chosen to use the term Electronic Performance Support Systems (EPSS) throughout this article, since that term has gained some popularity in the literature and is inclusive of a variety of forms of technology-based performance support. The article concludes with some general observations about areas of needed research and development and suggests implications of EPSSs for training developers and for the groups developing competency standards for instructional designers.

What Is an Electronic Performance Support System?

Gery (1991) and others have defined an EPSS as a system to integrate a knowledge base, learning experiences, and guidance to provide individuals the ability to perform at a higher level in the workplace. Among other terms used are: Performance Support, Online Performance Support, Performance Support Tool, Performance Support Engineering, Performance Centered Design, and Embedded Support.

Vol. 40, No. 1, January-February, 2000, P. 38-44
ET magazine Website: **http://BooksToRead.com/etp**

The concept of performance support has been around since early cave dwellers modeled behavior for others to imitate. In fact, watching others, and even better, being able to ask them questions and obtain their feedback on performance is still a very effective way to become a better performer and has been incorporated into some scenario-based EPSSs (usually in the form of video segments). However, this level of personal support is expensive to provide, since often neither the learner nor the advisor are being productive during any help sessions. Additionally, as the workplace becomes increasingly complex, quick and accurate information processing becomes a competitive necessity. And, according to Winslow and Bramer (1994), as the rate of change accelerates, even experts may have to struggle to maintain their level of performance. With the now almost ubiquitous presence of the computer in the workplace, it becomes possible to provide a variety of new forms of support to aid worker performance and potentially reduce the amount of training they require. Performance support may be as simple as providing a structured template for a person providing rate quotes for various forms of telephone service, or as complex as an intelligent agent or "knowbot" searching for information on the Web to answer a question about the latest research findings for a medical researcher.

In her original definition of an EPSS, Gery (1991) included four elements: an information base, some form of advisor, tutorials, and tools to assist the user. As EPSS design has matured, information bases may now include multiple knowledge sources; advising may take on features of expert systems or artificial intelligence; tutorials may be extensive and contain contextual multimedia instruction; and the tools have become more sophisticated. There is every reason to believe these trends will continue as more experience is gained in creating EPSSs.

What Are Some Examples of EPSSs?

While individual definitions of what constitutes an EPSS vary, mine tends to be broad and encompassing. Thus, it includes everything from the simple telephone rate quote template mentioned earlier to EPSSs for operators of "high tech" military communications systems based on expert systems and artificial intelligence. This definition also includes systems that may temporarily (or permanently) bypass the actual worker by interacting directly with customers or clients of an organization. One example of the latter is a software package that allows the do-it-yourselfer to design a backyard deck and have the materials list automatically generated and priced by a lumber company. Another example is a Web-based decision aid to assist customers in deciding which copy machine, with which specific features, best matches their requirements. In each instance, a sales representative can then double-check the results and process the customer's order.

Computer application software designers are also beginning to include various forms of performance support that go well beyond the notoriously ineffective online documentation provided with application software. Herein lies an important message for those embarking on the design of an EPSS—simply putting existing documentation or other information online and making it searchable does not make for an effective EPSS. There is very little value added by unstructured online information and documentation systems; although doing so has saved a lot of trees. Information is not knowledge and knowledge is what is required for performance. Glimpses of what becomes possible in the computer environment are the "wizards" now being shipped with some word processors and integrated software applications. Some of these wizards will prompt the user, who for example invokes the table function within a word processor to ask if help is wanted in formatting the table's layout. Allowing users to select from pre-existing formats (provided by the software creators or locally created and stored) provides another form of valuable help, as does the ability of software to dynamically exchange data and update spreadsheets when a database is modified.

Vol. 40, No. 1, January-February, 2000, P. 38-44
ET magazine Website: **http://BooksToRead.com/etp**

One example of an EPSS that I find compelling is work done by the Veterans Administration in the United States to overcome the incredibly poor design of its current financial management system (FMS) (Hawkins, Gustafson, & Nielsen, 1998). The FMS is based on a mainframe, line editing, data entry format commonly used in the 1970s with dumb terminals. The user sees only cryptic codes for the various types of data to be entered, and the screen layout does not correspond to the format of the printed form. While new software for the system is under design, it is not expected to be operational for several more years, so the present system must continue to be used in the meantime. Recently developed technology allows desktop computers to display the terminal generated display with an overlay of "clickable" buttons and a pull-down menu that provides users with various forms of help, including explanations of the codes, samples of the type of data associated with the codes, document status information, a list of frequently asked questions, hyperlinks to other VA information sources, and brief tutorials. All of these features are under user control.

Performance support systems are also impacting how instructional design is performed. Systems to support design and development of education and training programs range from simple templates and decision aids to expert system based software. An example of the former was developed by Gustafson and Reeves (1990) and consisted of an instructional design model with "tools" to support each of the model's elements. Accompanying each tool is an explanation of how and when to use it, along with completed samples resulting from its use. A somewhat related form of EPSS was developed for NCR Corporation as reported by Jury, Gustafson, & Reeves (1993).

An example of expert system based software that will automatically generate executable code for computer-based training modules was developed by Merrill (1993). In this system, a subject matter expert enters content information directly into a computer via a structured interview. The software then generates the actual training using a knowledge base and decision rules contained in the system. Although to date Merrill's system can accommodate only a limited array of declarative and procedural knowledge learning outcomes, its developers believe it can play a significant role in increasing the efficiency, and perhaps the effectiveness, of these forms of instruction.

An additional source of examples of EPSSs is at a Website—http://www.epss.com—that is dedicated to the topic. This site also contains a variety of other types of useful information about performance support.

Why Should Instructional Designers and Trainers Be Interested in EPSSs?

Design and use of job aids have a long history as a means of enhancing job performance (e.g., Milheim, 1997; Rossett, 1991; Zemke, 1982). Well-designed job aids, ranging from pilot checklists, to decision aids, to assist sales personnel in matching specific products to customer requirements, have consistently demonstrated their ability to improve on-the-job performance. They have also been shown to reduce training time, promote transfer from the learning environment to the workplace, and increase worker satisfaction. In some cases, EPSSs represent simply placing conventional job aids into an electronic environment. However, powerful desktop and portable personal computers, especially those that are networked and have multimedia capability, create a potential for assisting workers in ways undreamed of only a few years ago. The ability for workers to quickly tap large and dynamic databases, view high-quality graphics, see and hear motion video, receive context-specific help, and have ready access to experts, requires thinking well beyond the bounds of traditional job aids.

Increasingly, organizations are showing an interest in EPSSs as a means of enhancing worker performance, either to augment (and perhaps shorten) training, or as a complete substitute for conventional "off-the-job" training. This interest is driven by a desire to promote quality performance by the most effective and efficient means possible. Thus, from the human resource development perspective, the question becomes "what are the respective roles of job design, training, and EPSS and how do they complement each other in promoting high quality performance?"

Instructional designers and trainers who ignore these developments do so at their own risk. No less a "training" *guru* than Robert Mager (1996) made this point in a somewhat different way when he predicted that those who view their role in organizations as providing training will have a bleak future unless they began to focus on worker performance instead. His message was not that training would disappear, but that the *amount* and *type* of formal training is dramatically altered when one focuses on on-the-job performance. Ultimately, employers are interested in paying for performance not training. This is leading more of them each year to introduce EPSSs into the workplace (Raybould, 1995; *Training*, 1997).

What Are Some of the Major Design Considerations When Creating an EPSS?

There are at least seven different considerations when creating (or selecting) an EPSS:

- Black box/glass box objective
- Part-task/whole task support
- Embedded/linked/external connection

Vol. 40, No. 1, January-February, 2000, P. 38-44
ET magazine Website: **http://BooksToRead.com/etp**

- Self-contained/networked and shared work space
- User controlled/system controlled
- User/organization modified
- Static/dynamic system

Each is discussed separately below but clearly these are overlapping and contingent decisions. Also, how the EPSS will complement (if at all) other forms of performance support, such as training, on-the-job supervision, and mentoring, will affect the importance of each consideration and how it might be incorporated into the EPSS.

Black Box/Glass Box Objective

From the very beginning stages of its conception, EPSS designers must decide to what degree, if at all, its goal is to make users more knowledgeable and skilled. In many cases, it may be quite possible to create a support system that has as its sole objective getting the task done efficiently and correctly without making the user more competent. In the case of a phone rate quote EPSS, it is a black box design, since there is no desire to make the user even aware of how the variables for which they collect data are used to arrive at the customer quote. In fact, there may be proprietary or security reasons why workers would not be allowed to examine underlying algorithms, decision models, or data structures accessed by the support system.

In contrast, a glass box design has as its goal making these underlying rules, models, algorithms, and other structures readily available, and perhaps even requiring that the user examine them. Thus, the goal might be making the user more competent in a broader sense than just getting the job done correctly. Perhaps the goal would be to eventually wean users completely from the support system. Under these latter conditions, the EPSS can become a form of training, with its own learning objectives to accompany its job performance objectives. Glass box design can become quite challenging, if it goes beyond the level of "here is a bunch of background information you can access" to designing a planned learning environment for all users. For example, the tax preparation software commonly available in the U.S. prompts users to enter data and will then prepare and print the forms submitted to the government. In addition, the software contains extensive online "help," but mostly it consists of Government regulations and obscure explanations of specific sections of the tax code. Clearly, this help is not intended to make one skilled in understanding tax law. In that sense, it is a useful, but nonetheless a blackbox, EPSS. In contrast, one can conceive of a support system that would provide very specific guidance and feedback to users and even include formal learning activities and assessment instruments to assure they were achieving the learning outcomes. A glass box tax preparation support system could be created by one of the companies that provides tax services to large numbers of clients. It would then have the dual purposes of guiding employees in preparing clients' tax returns, while also becoming more expert in tax laws and regulations.

So, the bottom line is, to what degree will the support system being designed incorporate any glass box elements, and will their use be optional or required? Of course, this issue also relates to the question of what other training will be provided and how it complements the support system. Early clarification of these objectives will avoid many problems later in the design and development process.

Part-Task/Whole Task Support

Although some workplace tasks are very simple, many are lengthy, complex, and perhaps highly procedural. For the latter type of tasks, a decision must be made as to whether to support only parts of the overall task or the entire process, procedure, or decision. For example, instructional design is a complex process consisting of many individual tasks (e.g., task analysis, audience analysis, lesson design, etc.). GAIDA, an EPSS developed by the Armstrong Lab of the U.S. Air Force, is based on Gagné's nine events of instruction and only supports the design of individual lessons (Gettman, McNelly, & Muraida, in press). In contrast, *Designer's Edge*, a commercially available system from Allen Communication, claims to support "analysis, design, and evaluation of effective technology-based training" (Allen Communication, 1998).

Assuming multiple subtasks or part-tasks are to be supported, the decision must then be made as to whether that support will be integrated across related tasks or each will be supported separately. Passing data from task to task and setting flags for future entries are examples of integrating across tasks. Intelligent systems may in the future be able to complete parts of related tasks using rules and existing databases to provide an even higher level of support.

Embedded/Linked/External Connection

When designing an EPSS, decisions must be made as to how the support elements will be connected to the job task faced by the user. When embedded, all of the pieces of the EPSS are seamlessly integrated (at least from the user's point of view) and may require no action by the user to be invoked. For example in Merrill's expert system based EPSS (1993), the user, typically a subject matter expert with little or no expertise in instructional design, enters information as prompted by the system. After all entries are completed, the system will compile an executable computer-based lesson. All of the necessary "rules" and code-generating instructions are embedded in the system, and the user simply views the results.

In contrast, linked support is readily available, but usually requires some action on the part of the user to be accessed. Of course, the system also could initiate access to linked elements. Linked elements might include an external database maintained elsewhere or even a live person at a help desk accessible from within the support system. Linked resources have both advantages and disadvantages, a fact to which any Web designer will readily testify. Links allow access to the most current information, but that can also lead to confusion if the format or contents of the linked site have changed since the link was established. And in the case of "live" support, everyone knows help desk personnel display varying degrees of competence. Clearly, then, whether or not to link is a major design decision.

External connections to a support system require the user to exit the work task environment to obtain support. Examples would include going to a reference document, studying an online tutorial module, exiting to a Web browser to conduct a search, or walking across the hall to seek personal assistance. Exiting the work environment has the disadvantage of taking the worker off the job task, but does create opportunities to take advantage of a variety of types of help that may already exist in the organization, but do not readily lend themselves to being directly linked to the support system. Making external connections raises unique design challenges.

Self-Contained/Networked and Shared Work Space

Many support systems are designed to operate only on stand-alone computers. Others may operate on a network to take advantage of its resources, but still support only individual users. However, EPSSs are now being developed to support group activity (Malcom, 1998). Computer supported collaborative work permits teams to work either synchronously or asynchronously and to be widely separated geographically. A logical extension of the ability to share files and workspace is to provide other forms of support, such as decision aids, change-tracking, linking to common resources, and brainstorming. As Internet2 becomes a reality and supports full-motion video and 3D graphics, fully-functioning work teams "at a distance" become technologically feasible. However, designing EPSSs for shared work spaces represents a significant challenge, since, in addition to understanding the relevant process or product, the designer must also anticipate how individuals will actually perform *as a team.*

User Controlled/System Controlled

Whether the individual or the system will control how and when the support system operates is another design decision. Some help systems operate solely at the option of the user, others may contain a mix of user and system control, while still others may be completely under system control. The wizards which some application programs contain are examples of support that is completely under user control. Although some will prompt users by reminding them that assistance is available (e.g., "you appear to be making a table, would you like some help?), the decision is strictly up to users. Tax preparation software usually has a combination of user and system control. Users can ask for information and assistance, but in selected places the input or calculation process will not advance unless the correct type of data is entered or when nothing has been entered in selected fields. The phone rate EPSS is completely under system control, with each successive data entry point prompted and the range of acceptable entries actively monitored, with correction required if the data are unacceptable. Another example of system control is Merrill's expert system instructional design software (1993).

Implications for the design process include making initial decisions about who controls what, determining how and when to invoke any support, and estimating the effect on user attitudes of having the system initiate or control the amount and type of assistance. Letting the user set the level of system control partially addresses the user attitude issue, but then the designer is faced with the question of how much of what type of control is passed to the user and its possible negative impact on performance.

User/Organization Modified

Another design decision is whether the user will be able to tailor or modify the support system, or if this function is reserved solely to the organization. Local modification has the advantage of making the system appear to be user-friendly. Being able to customize displays, shut off system prompts and reminders, enter one's own cues, tips, and other unique aids, and modify the system to local and or changing conditions may make users more efficient and enhance their sense of ownership. The EPSS developed for the VA even allows users to input ideas, samples, and examples that others can access, with contributors being recognized for their contribution.

Of course, allowing user modification may have a negative effect if they disable critical assistance and proceed blissfully to make and perhaps compound errors. As the user's expertise increases, either the system or the user may have the capability to adjust the amount or type of support initiated. Finding the balance between user adaptation and system control, especially when users range from novice to expert, creates significant new design challenges.

Static/Dynamic System

Whether the support system will be static or dynamic is related to the issue of who can make modifications and when and how they are made. Support systems that can "learn" about the user or the task have great intuitive appeal (Laffey, 1995; Laffey, Tupper, Musser, & Wedman, 1998; Malcom, 1998), but their design represents a formidable undertaking. Further, if the nature of the changes affects how users interact with the system, users can become disoriented by being required to change well-ingrained patterns of behavior, and rearranging their electronic workspace. If the benefits of any changes do not clearly outweigh the "costs," then morale and performance are likely to suffer. A simple example of a dynamic system is tax preparation software that remembers your personal information, such as Social Security number, spouse's and children's names and ages, tax refunds, and types of financial accounts you reported the previous year, when you start work on the subsequent year's tax returns. Dynamic support systems might also be developed that track how one works and tailor themselves to those preferences. On the other hand, static systems have the advantages of stability of user interface, consistency of information storage and retrieval, and predictability of operation. They also are likely to be easier to design, implement, and maintain. Thus, the sponsor and designer need to decide very early whether the support system is to be static or dynamic since many structural elements are likely to be different for each design.

How Independent Are These Design Considerations?

Although the seven design considerations were discussed separately, in reality, decisions made about any one will create and eliminate options related to at least some of the others. Thus, I do not see the day any time soon when a decision tree or algorithm will be created to determine the balance among them. Rather, there are multiple tradeoffs to be made, and EPSS designers will need to be able to explain the options and consequences to clients and negotiate acceptable (and achievable) design parameters with them if they are to provide the most valuable service.

Vol. 40, No. 1, January-February, 2000, P. 38-44
ET magazine Website: **http://BooksToRead.com/etp**

How Does One Develop an EPSS?

Unfortunately, there is very little literature available that describes how people have actually designed and developed EPSSs. There are at least two reasons for this. First, in the commercial marketplace, proprietary self-interests dictate that many vendors not disclose the specifics of how they work. Second, the field of electronic performance support is relatively new, and procedures have not been well-developed and tested, as is now the case with conventional instructional design. There may also be a third reason in that some EPSS designers may be reluctant to talk about what they have done, since they are unable to clearly articulate specific and replicable procedures. Some of the more interesting support systems evidence a great deal of creativity and imagination that is not easily captured into procedural rules and is not based on empirical evidence. In essence, EPSS design is an immature technology about which there is much more to be learned.

Having said that, however, it is still possible to provide some guidance to would-be designers. First, many elements of classic instructional design can play a major role. Task analysis, audience analysis, environmental analysis, prototype development, and formative evaluation tools and techniques are directly applicable when an EPSS is being created for an existing process, such as supporting financial transactions in a bank. Traditional design processes probably work best when the performance environment is stable, the information base and any accompanying rules and decision algorithms are known and stable, and the electronic environment into which it will be installed is uniform and stable.

A second possible approach to EPSS development is rapid prototyping. This approach has its roots in application software design; especially for user interface design (Hix & Hartson, 1993). Rapid prototyping (RP) uses a highly interative cycle of prototype development and user tryout. Heavy emphasis is placed on user input and not making an early commitment to any specific design elements (Moonen, 1996; Tripp & Bichelmeyer, 1990). Competing ideas may be tested in primitive form before decisions are made, and often elements from different prototypes are combined in later ones. Rapid prototyping may work best when the process or product it is to support is still emerging or the client wants something “different,” but is unable to specify what that means. Creativity can blossom during RP. The result can be a breakthrough application or a train wreck; and managing RP remains an art form that should not be attempted by the faint of heart.

Vol. 40, No. 1, January-February, 2000, P. 38-44
ET magazine Website: **http://BooksToRead.com/etp**

A third possible approach to EPSS development is concurrent engineering (CE). Concurrent engineering is now being used by numerous companies when they embark on new product development (Keys, 1992; Watson, 1993). Keys defines CE as simultaneous and integrated design and development of a product, including everything from conceptual design, to manufacturing, to marketing, to distribution. Concurrent engineering is based on assembling an interdisciplinary team, typically consisting of design and manufacturing engineers, repair and maintenance personnel, sales and marketing representatives, product managers, and—very importantly—user representatives. This team works together until the product is released, and even beyond if the situation warrants. One of the more interesting examples of the impact of CE on the very survival of a company is Team Taurus (Watson, 1993), that designed the automobile generally considered to have revived the fortunes of Ford Motor Company.

Although I have not located any literature describing the use of concurrent engineering to develop an EPSS, I remain convinced it is being used by some software development companies. Binkert, Lippit, and Spannaus (1998) described a concept called "Full Service Performance Improvement" that appears to contain most of the attributes of CE, but the specific role of prototypes is not clearly described. It is also worthy of note that I have found no mention of trainers or instructional designers as team members in any of the CE literature. However, it seems likely that buried somewhere in the corporate world are CE teams with members of the EPSS community planning how to support new products; especially those being designed to be used in electronic environments.

Although three different approaches to EPSS design have been described, in reality, some blending across them seems likely. For example, employing traditional ID to determine performance requirements and then using RP to design at least parts of the EPSS taps the strengths of each approach. Interestingly, this blending of approaches is what has been found in how much instructional design is actually practiced, although it is not how practitioners are taught or how the process is typically described in the literature (Rowland, 1992; Visscher-Voerman, Gustafson, & Plomp, in press). Similarly, CE team members often use RP to test ideas while they are still on the drawing board by using expert opinion and during prototype testing with end users. Exactly how selected instructional design tools and techniques fit into this framework provides an exciting area for experimentation.

Implications of EPSS for Instructional Design Practitioners

It is also seems clear that many of the exciting opportunities to enhance performance (and learning) will involve EPSSs and other forms of environmental modification. The design and of these modalities will require alternative (or mixed) methods such as RP and CE. Those in the instructional design field who desire to explore and work in these exciting areas will need to acquire new knowledge and skills and change their perspective from training to performance. This latter change may be the hardest one to make for those who have experienced success and recognition, but a substantial part of the future belongs to those who can make this change.

Another implication for design practitioners is the need to think more about the difference between *tool use* and *tool building.* Much traditional instructinal design has taken place in a "low tech" world, with the designer's major tool being the word processor. As the pace of change accelerates and the workplace becomes more complex, instructional design, in whatever form it takes, will require greater use of performance support tools, such as those created for NCR Corporation (Jury, Gustafson, & Reeves, 1993). And, at another level, more tool builders are needed to create the EPSS software that others can use to more efficiently and effectively create training programs.

Thinking even beyond EPSSs for training development, the next step is to create EPSSs to create EPSSs. Clearly the opportunities have never been greater for those of us in the design business to become tool builders. Now the question is whether we will seize the moment or take the more comfortable path of continuing to do what we now do well and leave the exploration to others. □

References

Allen Communication. (1998). See Website at: http://www.allencomm.com/

Binkert, J., Lippit, L., & Spannaus, T. (1998). Full service performance improvement in an age of specialization. Paper presented at the IBSTPI Performance Improvement Workshop, University of Bergen, Norway.

Foster, E. (1997). See Website at: http://www.epss.com/lb/artonlin/artonlin.htm

Gery, G. (1991). *Electronic performance support systems.* Cambridge, MA: Ziff Institute.

Gettman, D., McNelly, T, & Muraida, D. (in press). The Guided Approach to Instructional Design Advising (GAIDA): A case-based approach to developing instructional design expertise. In J. van den Akker, N. Nieveen, & Tj. Plomp (Eds.), *Design methodology and development research in education and training.* Dordrecht: Kluwer Academic Publishers.

Gustafson, K., & Reeves, T. (1990). IDioM: A platform for a course development expert system. *Educational Technology, 30*(3), 19–25.

Gustafson, K., & Branch, R. (1997). *Survey of instructional development models* (3rd ed.). Syracuse, NY: ERIC Clearinghouse on Information Resources, Syracuse University.

Hawkins, C., Gustafson, K., & Nielsen, T. (1998). Return on investment (ROI) for electronic performance support systems: A Web-based system. *Educational Technology, 38*(4), 15–21.

Hix, D., & Hartson, H. (1993). *Developing user interfaces: Ensuring usability through product & Process.* New York: John Wiley & Sons.

Jury, T., Gustafson, K. L., & Reeves, T. C. (1993). *Quality information product process performance support system (QIPP-PSS).* Electronic performance support system for Information Products Division of NCR Corporation, Dayton, OH.

Keys, L. (1992). Concurrent engineering for customer, industrial products, and government systems. *IEEE Transactions on Components, and Hybrids, and Manufacturing Technology, 15*(3), 282–287.

Laffey, J. (1995.). Dynamism in electronic performance support systems. *Performance Improvement Quarterly, 8*(1), 31–46.

Laffey, J., Tupper, T., Musser, D., & Wedman, J. (1998). A computer-mediated support system for project-based learning. *Educational Technology Research and Development, 46*(1), 73–86.

Mager, R. F. (1996). Morphing into a 21st century trainer. *Training, 33*(6), 47–54.

Malcom, S. (1998). Where EPSS will go from here. *Training, 35*(3), 64–68.

Merrill, M. D. (1993). An integrated model for automating instructional design and delivery. In J. M. Spector, M. C. Polson, & D. J. Muraida (Eds.), *Automating instructional design: Concepts and issues* (pp. 147–190). Englewood Cliffs: Educational Technology Publications.

Milheim, W. (1997). Instructional design issues for electronic performance support systems. *British Journal of Educational Technology, 28*(2), 103–110.

Moonen, J. C. M. M. (1996). Prototyping as a design method. In Tj. Plomp & D. P. Ely (Eds.), *International encyclopedia of educational technology* (2nd ed., pp.186–190). Cambridge, UK: Pergamon.

Raybould, B. (1995). Performance support engineering: An emerging EPSS development methodology for enabling organizational learning. *Performance Improvement Quarterly, 8*(1), 7–22.

Richards, B. F. (1988). Intelligent job aids for professionals: Lessons learned from medicine. *Performance Improvement Quarterly, 1*(1).

Rossett, A. (1991). Job aids in a performance technology world. *Performance and Instruction, 30*(5), 1–6.

Rowland, G. (1992). What do instructional designers actually do? An initial investigation of expert practice. *Performance Improvement Quarterly, 5*(2), 65–86.

Spector, J. M., Polson, M. C., & Muraida, D. J. (Eds.) (1993). *Automating instructional design: Concepts and issues.* Englewood Cliffs: Educational Technology Publications.

Training. (1997). Industry report: Technology Training. *Training*,

34(10), 67–75.

Tripp, S.D., & Bichelmeyer, B. (1990). Rapid prototyping: An alternative instructional design strategy. *Educational Technology Research and Development, 38*(1), 31–44.

Visscher-Voerman, I., Gustafson, K., & Plomp, T. (In press). Educational design and development: An overview of paradigms. In J. van den Akker, N. Nieveen, & Tj. Plomp (Eds.), *Design methodology and development research in education and training.* Dordrecht: Kluwer Academic Publishers.

Watson, G. (1993). *Strategic benchmarking: How to rate your company's performance against the world's best.* New York: John Wiley and Sons.

Winslow, C., & Bramer, W. (1994). *Future work: Putting knowledge to work in the knowledge economy.* New York: Free Press.

Zemke, R (1982). *Figuring things out.* Reading, MA: Addison-Wesley.

Kent L. Gustafson is Professor, Instructional Technology Department, University of Georgia, Athens, Georgia (e-mail: kgustafs@coe.uga.edu).