Systems that Help People Get Work Done

by Peter R. Hybert

Your company's information system is more than computer hardware and software. Manuals, training courses, meetings, newsletters, and measurement data tracking are all part of the process "information" that supports the work taking place in the business. If employees working in the business can be called "performers" because they *perform* process tasks, then this overall information system can be considered a human performance *support* system because it enables or helps people to do their jobs.

The mission of the human performance support system is to provide the knowledge, skills, and information to the points in the process where they are needed for completing process tasks. Improvements to the human performance support system should enhance business results (e.g., quality, productivity, or cost reduction), keeping in mind that the benefits may actually be realized elsewhere in the business. For instance, using a computer-aided design system may actually *increase* the initial design cycle time but will save much more downstream by reducing the cost and problems of transferring the design to manufacturing.

The human performance support system consists of a pipeline or distribution system for the knowledge, skills, and information along with the content of that pipeline. Figure 1 illustrates where the system fits within the work environment.

The process determines the repertoire of knowledge and skills needed by the performers. A bank teller would need customer service skills, basic math, knowledge about the policies or laws that limit the transactions they can and cannot perform, among other things. The repertoire needs to be resident in the *performer*—they cannot reference a manual or computer and still perform the task suitably. But there is information that he/she would not be expected to learn, such as specific account numbers and balances for individual customers. If this information is needed for a transaction, he or she needs a way to *reference* an information source as close to the point of need and as quickly as possible.

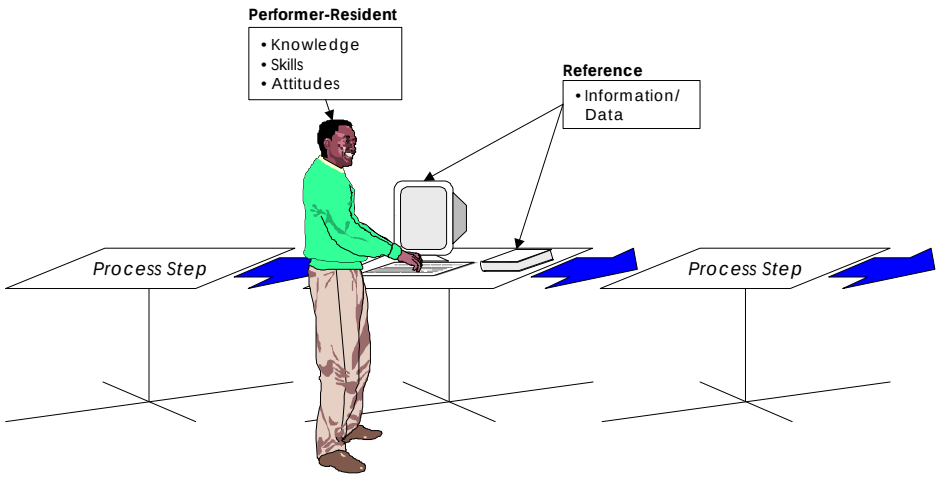


Figure 1: Human Performance Support System within Work Environment

What is Wrong with the Average Human Performance Support System?

There are a number of ways that people obtain the knowledge, skills, and information that support them in performing their work. Many are provided by a formal delivery system (such as training, computer databases and tools, or reference manuals), while others may be created by individual performers (such as the "Post-Its®" on the computer keyboard). These vehicles of information and learning are normally not designed, developed, or maintained as an integrated *system* but evolve as individual products managed by different organizations—usually there are numerous gaps, inconsistencies, and overlaps between them that introduce errors, delays, waste, and frustration into the work processes.

And, while the systems that collect, store, and distribute knowledge, skills, and information are often poorly positioned to support the *current* business processes, they are now being expected to support processes that are being re-engineered for a *future* business environment. This future business environment, in which "knowledge work" is the core of the "information economy," will place even greater demands on the human performance support system.

In addition, the lines between the various elements of the system traditionally follow organizational boundaries. This has always led to problems, since the multiple deliverables are aimed at a shared audience and can cover the same content. But it becomes even more critical as the lines between training, documentation, and on-line help are blurred as performance support is increasingly delivered via computer. Software developers tend to focus on how the computer will process the data rather than on how people will be using the software. Training specialists, on the other hand, often do not understand the work process or technology sufficiently to participate in and even influence the software design decisions.

What Is Needed?

An integrated product development approach can contribute a great deal toward a better fit between the work process and the software and other human performance support. Six key points are summarized below.

- Use a team-based approach.
- Focus on the work process and business results measures.
- Design it all as a system.
- Allocate ownership by vehicle and contents.
- Design for the user.
- Prototype (and test) the elements together.

Use a Team-based Approach

Using a cross-functional team—including user representatives, training and documentation professionals, human factors professionals, as well as IS professionals—to design processes and information support is not uncommon in many environments. However, the human performance support specialists frequently join the team late in the process when they can add more value early in the requirements definition and process design phases.

Another common problem is that many team leaders do not know how to facilitate the team through the planning needed at the beginning of a project. As a result, the team members become confused or discouraged about their roles. Team leaders need to lead a discovery process in which the team members figure out the project and their deliverables together so that all the expertise areas can contribute. For this to happen, the team leader needs to be a *facilitator* instead of a technical expert. The planning effort should focus on understanding and documenting the tasks to be performed, the knowledge and skills that need to be performer-resident, and the information that needs to be referenced by the performer.

Focus on the Work Process and Business Results Measures

Every process redesign, improvement, or automation effort ought to be done for a business rationale or you are potentially wasting the shareholders' money. That business rationale should be able to be translated to measures that impact some combination of customer satisfaction, competitiveness, market share and/or business volume growth, cost reduction, or employee satisfaction. These measures can be deployed to parts of the system and tracked to verify project success. Will the new ordering system reduce the cycle time for processing an order? Will it reduce errors? Will this help us win more customers? Defining these targets and then baselining today's performance will give the team reference points for measuring success as the system is implemented.

Design It All as a System

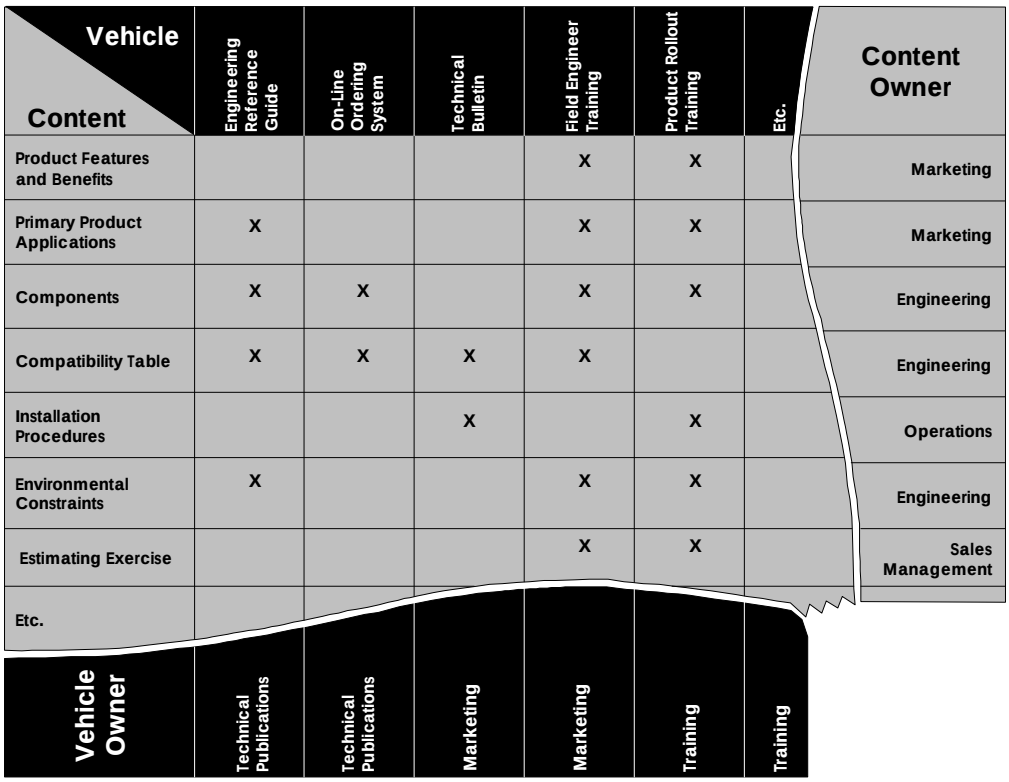
Development of large-scale systems includes a system architecture that defines the elements of the system and the relationships between them. The human performance support system should be no different. Knowledge, skills, and information requirements of the performers should be allocated to the various components of the system. This can best be done with the team so that overlaps, gaps, and synergies can be identified, otherwise you could incur redundant development costs.

The systems architecture should, at the least, specify the mission, user(s), boundaries (what it does and does not include), interfaces with other elements, and life cycle for each component of the system. A training program may incorporate another support element, such as a user guide, and this should be defined as part of the system architecture. That same training program may be intended to exist only during the rollout of the new system while some of its contents may become part of a new employee course for future audiences—these plans should also be described in the system architecture. The system architecture allows the team to clarify their "turf" and also see how they can best work together.

Allocate Ownership by Vehicle and Contents

Since in a typical business a number of organizations may "own" the various deliverables that make up the human performance support system, ownership roles need to be clear to the entire team. Clarity requires separate identification of the ownership of the *vehicle* that delivers the content, which means responsibility for development, production, distribution, and maintenance, versus ownership of the *content* itself, which means responsibility for the accuracy of the information contained.

For example, the content of an installation procedure should be defined by the software developer but might appear in a start-up manual, on-line help, and a "quick reference" card. The software developer needs to sign off on the accuracy of the information, but the documentation developer needs to be responsible for deciding where and in what format this information will appear in his particular document to best support the tasks being performed. (See Figure 2 for a matrix that illustrates how the ownership could be mapped for a team.)



Vehicle / Content	Engineering Reference Guide	On-Line Ordering System	Technical Bulletin	Field Engineer Training	Product Rollout Training	Etc.	Content Owner
Product Features and Benefits				X	X		Marketing
Primary Product Applications	X			X	X		Marketing
Components	X	X		X	X		Engineering
Compatibility Table	X	X	X	X			Engineering
Installation Procedures			X		X		Operations
Environmental Constraints	X			X	X		Engineering
Estimating Exercise				X	X		Sales Management
Etc.							
Vehicle Owner	Technical Publications	Technical Publications	Marketing	Marketing	Training	Training	

Figure 2: Example Matrix for Mapping Ownership

Design for the User

"User friendly" is almost a cliché today, but too often the focus is strictly on the interface. While the interface is critical, there also needs to be a focus on the overall job performance of the user. Are we expecting them to perform too many steps? Are we expecting them to remember or know too much information? Can they use our system quickly enough to get the job done under typical working conditions? Involving users can help, but they may not always know what can be done through software, training, or even process redesign. Most design teams will benefit from even limited hands-on experience with the process and tasks in the user's environment.

For example, in a new product development effort, the design for a networked digital control device called for the field engineer to determine the size and type of transformer(s) needed for each installation depending on the number of devices on the network and the distance between them. This was a new and unfamiliar task to the existing audience, and it required several steps to complete the necessary formulae—it would require a great deal of human performance support to minimize the risk of error.

Walking through the process led the team to modify the product design to work with two standard transformer sizes. Instead of training the engineers on transformer sizing, the human performance support consisted of a simple table for making the selection. In addition, a "failsafe" was put into the ordering system so that it would kick out orders that did not fit the established parameters or did not include transformers at all. This solution resulted in lower overall operating cost and error rate for the final process—more than adequate return for the effort.

Prototype (and Test) the Elements Together

Much has been written about the benefits of rapid prototyping, and these benefits pertain to the development of the human performance support system as well. However, just as a prototyping strategy should be designed into the software development process for specific reasons, such as assessing user reaction to the interface, human performance support system prototyping should be done for specific purposes. Individual components of the human performance support system will require testing on their own if technical boundaries are being pushed or the system is very different from what people are accustomed to today.

More importantly, the software and human performance support system should be tested with users as a single entity that redefines their performance environment. Problems, questions, errors, etc. can be identified, and the team can decide together on the best way to address them. It may be better to make a modification to the software than to "beef up" the training or vice versa.

Conclusion

As the line continues to blur between business process design, software application development, and electronic and non-electronic "infoware," it will become increasingly important to use a team approach to designing the entire system to optimize human performance for business results. Training, documentation, human factors, and IS professionals will need to find ways of working together effectively throughout the systems development process.

Author's note: This article was originally written in 1995 and was submitted for publication to a journal on systems. It was rejected but I found the editor's note amusing. I'm paraphrasing but the message was that "everybody probably knows this…though maybe it wouldn't hurt to hear it again because hardly anyone does things this way." I agreed with the second half of the statement (which is why I wrote the article in the first place!) but the notion that it wasn't a new message took the wind out of my sails. In retrospect, I wish I had made the effort to follow-up with that journal or another.