Collaborative Information Environments for LifeLong Learning in Communities

Gerry Stahl, Gerhard Fischer, Jonathan Ostwald
Center for LifeLong Learning and Design
University of Colorado at Boulder
Boulder, CO 80309-0430 USA
+1 303 492 3912
{gerry, gerhard, ostwald} @cs.colorado.edu

ABSTRACT
Computer-based design environments for skilled domain workers have recently graduated from research prototypes to commercial products, supporting the learning of individual designers. Such systems do not, however, adequately support the collaborative nature of work or the evolution of knowledge within communities of practice. If innovation is to be supported within collaborative efforts, *domain-oriented design environments* must be extended to become *collaborative information environments,* capable of providing effective community memories for managing information and facilitating lifelong learning within constantly evolving community work contexts.

Keywords
Collaborative information environment, domain-oriented design environment, lifelong learning, organizational learning, community memory, community of practice.

COMPUTER SUPPORT FOR INDIVIDUALS

The Need for LifeLong Learning

The creation of innovative artifacts in our complex world–with its refined division of labor and its flood of information–requires continual learning. Learning can no longer be conceived of as an activity confined to the classroom and to an individual's early years. Learning must continue while one is a worker, a citizen and an engaged adult for several reasons:

- Innovative tasks are ill-defined; their solution involves the learning of information that could not have been predicted [28].
- There is too much knowledge, even within specific subject areas, for anyone to master it all in advance or on one's own [9].
- The knowledge in many domains evolves rapidly and often depends upon the context of one's task situation, including one's support community [6].
- Frequently, the most important information has to do with a work group's own structure and history, its standard practices and roles, the details and design rationale of its local accomplishments [24].
- People's careers and self-directed interests require various new forms of learning at different stages as their roles in communities change [17].
- Learning—especially collaborative learning—has become a new form of labor, an integral component of work and organizations [36].

The contemporary need to extend the learning process from schooling into organizational and community realms is known as *lifelong learning.* Our past research explored the computer support of lifelong learning with *domain-oriented design environments* (DODEs). This paper argues for extending that approach to support collaborative work with *collaborative information environments* (CIEs).

Section 1 illustrates how computer support for lifelong learning has already been developed for individuals such as designers. It argues, however, that DODEs that deliver domain knowledge to individuals when it is relevant to their task are not sufficient for supporting innovative work within collaborative communities. Section 2 sketches a theory of how software productivity environments for design work by individuals can be extended to support lifelong learning in collaborative work settings known as *communities of practice*. Section 3 provides a suggestive *scenario* of a CIE being used by a community of computer network managers. Finally, Section 4 touches on a set of critical CSCW issues concerning the design of CIEs.

Domain-Oriented Design Environments

Many creative work tasks can be conceived of as *design* processes: elaborating a new idea, planning a presentation, balancing conflicting proposals or writing a visionary report. While designing can proceed on an intuitive level based on tacit expertise, it periodically encounters breakdowns in understanding where explicit reflection on new knowledge may be needed [29]. Designing entails learning.

For the past decade, we have explored the creation of DODEs to support workers as designers. These systems are *domain-oriented*: they incorporate knowledge specific to the work domain [7]. They are able to recognize when a *breakdown* in understanding may occur and can respond to it with appropriate information [8].

To go beyond the power of pencil-and-paper representations, software systems for lifelong learning must "understand" something of the tasks they are supporting. This is accomplished by building into the system knowledge of the domain, including design objects and design rationale. A DODE typically provides a computational workspace within which a designer can construct and represent an artifact. Unlike a CAD system, in which the software only stores positions of lines, a DODE maintains a *representation* of objects that are meaningful in the domain. For instance, an environment for local-area network (LAN) design (our primary example in this paper) allows a designer to construct a network design by arranging items from a palette of icons representing workstations, servers, routers, cables and other devices from the LAN domain.

A DODE can contain domain knowledge about constraints, rules of thumb and design rationale. It uses this information to respond to a current design state with active advice. Our systems used a mechanism we call *critiquing* [13]. The system maintains a representation of the semantics of the design situation: usually the two-dimensional location of palette items representing design components. Critic rules are applied to the design representation. When a rule "fires," it posts a message alerting the designer that a problem might exist. The message includes links to information such as design rationale associated with the critic rule.

For example, a LAN design DODE might notice that the length of a cable in a network plan exceeds the specifications for that type of cable, that a router is needed to connect two subnets or that two connected devices are incompatible. At this point, the system could signal a possible design breakdown and provide domain knowledge relevant to the cited problem. The evaluation of the situation and the choice of action is up to the human designer, but now the designer has been given access to information relevant to making a decision [9].

Research Prototypes

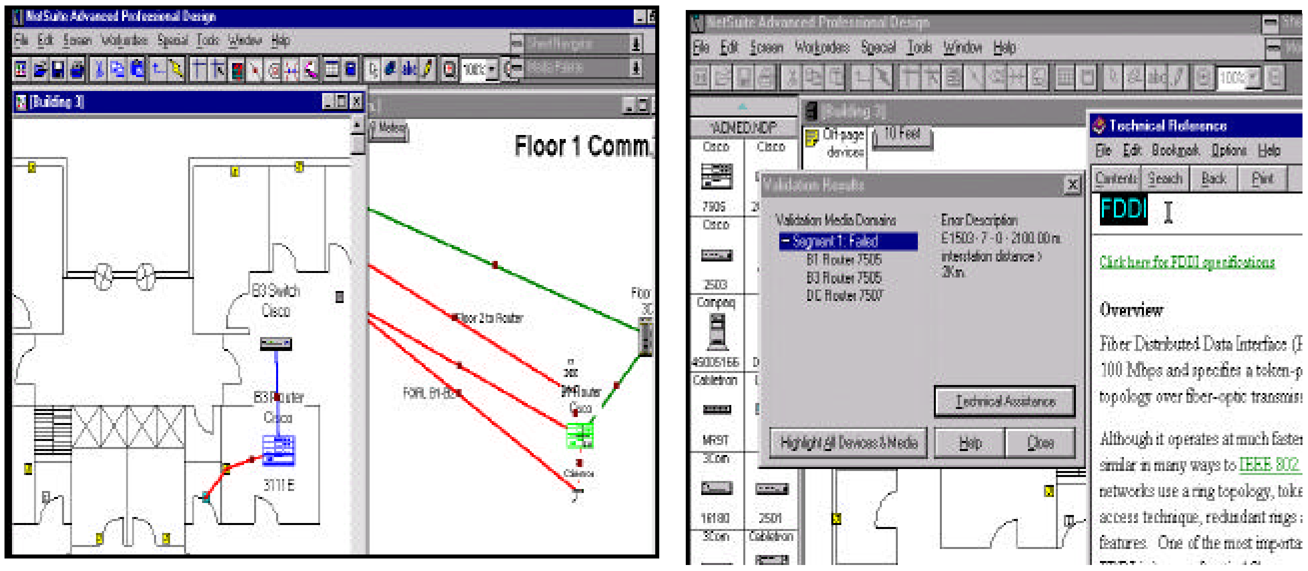
We have explored the utility of DODEs in a variety of domains. In a system for critiquing Lisp programs, we found that a software development environment could provide active suggestions backed up by rationale, but that it was limited if there was no representation of the designer's task (e.g., what the Lisp program was designed to do) [11]. An early DODE for constructing user interface designs provided a construction kit consisting of a palette of widgets so that the evolving design could be represented in a form that the software could analyze [10]. This led to DODEs for architectural design [13], including components for design rationale, end-user modifiability of palette items and design specifications. Finally, a series of prototypes for LAN design incorporated simulation and support for long-term, indirect communication (design history or community memory) [14].

These systems—and others in diverse domains like lunar habitat design [31] or voice dialog design [33]—included features to support collaborative design as well as the work of individuals. They investigated a variety of mechanisms for: seeding environments with catalogs of paradigmatic designs as starting points, allowing designers to share successful artifact designs, storing annotated design histories and archiving email design discussions They provided for flexible delivery of this information, tailored to individual design contexts. Such collaborative features will be discussed later in the paper.

A Commercial Product

Many of the ideas of our DODEs are now appearing in commercial products. In particular, there are shrink-wrap environments for designing LANs. As an example, consider NetSuite, a highly rated system that illustrates current best practices in LAN design support [21]. This is a high-functionality system for skilled domain professionals who are willing to learn to use its rich set of capabilities (see Figure 1). NetSuite contains a wealth of domain knowledge. Its palette of devices numbers over 5,000, with more downloadable from the vendor every month. Each device has associated parameters defining its characteristics, limitations and compatibilities—domain knowledge used by the critics that validate designs.

In NetSuite, one designs a LAN from scratch, placing devices and cables from the palette. As the design progresses, the system validates it, critiquing it according to rules and parameters stored in its domain knowledge. The designer is informed about relevant issues in a number of ways: lists of devices to substitute into a design are restricted by the system to compatible choices, limited design rationale is displayed with the option of linking to further details and technical terms are defined with hypertext links. In addition to the construction area there are LAN tools, such as an automated IP address generator and utilities for reporting on physically existing LAN configurations. When a design is completed, a bill-of-materials can be printed out and an HTML page can be produced for display on the Internet. NetSuite is a knowledgeable, well constructed system to support an *individual* LAN designer.



Validation Media Domains
Segment 1: Failed
B1 Router 7505
B3 Router 7505
DC Router 7507

Error Description
E1503 - 7 - 0 - 2100.00 m interstation distance > 2Km.

Click here for FDDI specifications

Overview

Fiber Distributed Data Interface (F
100 Mbps and specifies a token-p
topology over fiber-optic transmis

Although it operates at much faste
similar in many ways to IEEE 802
networks use a ring topology, toke
access technique, redundant rings
features. One of the most importa

Figure 1. Two views of NetSuite*. In the left view, parts of the network are being connected in physical and logical representations. In the right view, the system has noted that a cable length specification for a FDDI network has been exceeded in the design and the system has delivered information about the specification and affected devices.*

The Need to Go Further

Based on our understanding of organizational learning [23] and our investigation of LAN design communities, we believe that in a domain like LAN management no closed system will suffice. The domain knowledge required to go beyond the functionality of NetSuite is too open-ended, constantly changing and dependent upon local circumstances. The next generation of commercial DODEs will have to support *extensibility* by end-users [20; 27] and *collaboration* within communities of practice. While a system like NetSuite has its place in helping to design complex networks from scratch, most work of LAN administrators involves on-going management tasks. We conducted informal ethnographic studies and found that most time is spent extending existing networks, debugging breakdowns in service and planning for future technologies.

Many LAN management organizations rely on home-grown information systems because they believe that critical parts of their local information are unique. A community of practice has its own ways of doing things. Generally, these local practices are understood tacitly and are propagated through apprenticeship [17; 24]. This causes problems when the old-timer who set things up is gone and when a newcomer does not know who to ask or even what to ask. A *community memory* [1; 18] is needed that captures local knowledge when it is generated (e.g., when a device is configured) and delivers knowledge when needed (when there is a problem with that device) without being explicitly queried.

The burden of entering all this information in the system must be distributed among the people doing the work and must be supported computationally to minimize the effort required. This means that the software environment must be thoroughly interactive so that users can easily enter data and comments. The information base should be seeded with basic domain knowledge so that users do not have to enter everything and so that the system is useful from the start. As the information space grows, there should be ways for people to restructure it so that its organization and functionality keep pace with its evolving contents and uses. DODEs must be extended to support processes of "seeding, evolution and reseeding" [12] of information within communities of practice, rather than being designed based on a view of designers working in isolation with relatively static domain knowledge.

SUPPORTING LEARNING IN COMMUNITIES

Communities of Practice

All work within a division of labor is social [19]. The job that one person performs is also performed similarly by others and relies upon vast social networks. Work is defined by *social practices* that are propagated through socialization, apprenticeship, training, schooling, and culture [15; 3], as well as by explicit standards. Often, work is performed by cooperating teams that form *communities of practice* within or across organizations [4].

Our interviews showed that computer network managers at our university work in concert. They need to share information about what they have done and how it is done with other team members and with other LAN managers elsewhere. For such a community, information about their own situation may be even more important than generic domain knowledge [24]. Support for LAN managers must provide community memory about how individual local devices have been configured as well as offer domain knowledge about standards, protocols and compatibilities.

Communities of practice can be co-located within an organization (e.g., at our university) or across a discipline (e.g., all directors of university networks). Before the World Wide Web existed, most computer support for communities of practice targeted individuals with desktop applications. The knowledge in the systems was mostly static domain knowledge. With intranets and interactive web sites, it is now possible to support distributed communities and also to maintain evolving information about local circumstances and group history.

Memories for Communities of Practice

Human and social evolution can be viewed as the successive development of increasingly effective forms of *memory* for learning, storing and sharing knowledge. Biological evolution gave us episodic, mimetic and mythical memory; then cultural evolution provided oral and written—external and shared—memory; finally modern technological evolution generates digital (computer-based) and global (Internet-based) memories [5; 22].

At each stage, the development of hardware capabilities must be followed by the adoption of appropriate skills and practices before the potential of the new information technology can be realized. External memories, incorporating symbolic representations, facilitated the growth of complex societies and sophisticated scientific understandings. Their effectiveness relied upon the spread of literacy and industrialization. Similarly, while the proliferation of networked computers ushers in the possibility of capturing new knowledge as it is produced within work groups and delivering relevant information on demand, the achievement of this potential requires the careful design of information systems, software interfaces and new work practices. Computer-based *community memories* must be matched with new social structures that produce and reproduce patterns of organizational learning.

Community memories are to communities of practice what human memories are to individuals. They make use of explicit, external, symbolic representations that allow for shared understanding within a community [25]. They make organizational learning possible within the group.

The Process of Organizational Learning

The ability of individual designers to proceed based on their tacit existing expertise [26] periodically breaks down and they have to rebuild their understanding of the situation through explicit reflection [29]. This reflective stage can be helped if they have good community support or effective computer support to bring relevant new information to bear on their problem. When they have comprehended the problem and incorporated the new understanding in their personal memories, we say they have learned. The process of *design* typically follows this cycle of breakdown and reinterpretation (see Figure 2, cycle on left) [30].

A similar process takes place at the group level, involving

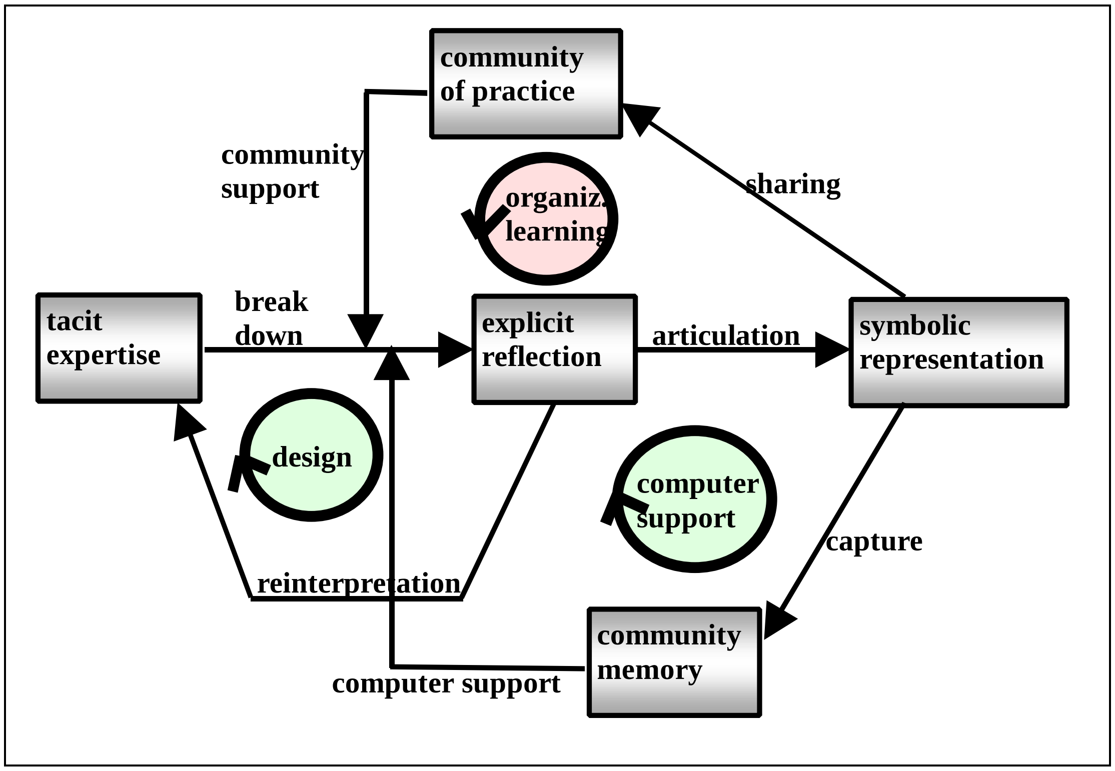


Figure 2. Cycles of design, computer support and organizational learning.

interaction between the community and its members. When design tasks take place in a collaborative context, the reflection results in articulation of solutions in language or in other symbolic representations. The articulated new knowledge can be shared within the community of practice. Such knowledge, learned by the community, can be used in future situations to help a member overcome a breakdown in understanding. This cycle of collaboration is called *organizational learning* (see Figure 2, upper cycle). The interaction of multiple personal perspectives and the collaborative articulation of shared perspectives makes innovation possible [2; 34].

Organizational learning can be supported by computer-based systems if the articulated knowledge is captured in a digital symbolic representation. The information must be stored and organized in a format that facilitates its subsequent identification and retrieval. In order to provide *computer support*, the software must be able to recognize breakdown situations when particular items of stored information might be useful to human reflection (see Figure 2, lower cycle) [31]. DODEs provide computer support for design by individuals. They need to be extended to CIEs to support organizational learning in communities of practice.

Extending the DODE Approach to CIEs

The key to active computer support that goes significantly beyond printed external memories is to have the system deliver the right information at the right time in the right way [13]. Somehow, the software must be able to analyze the state of the work being undertaken, identify likely breakdowns, locate relevant information and deliver that information in a timely and useful manner.

DODEs like NETSUITE and our older prototypes used *critics* based on *domain knowledge* to *deliver information* relevant to the current state of a *design artifact* being constructed in the design environment work space (see Figure 3, left).

One can generalize from the critiquing approach of these DODEs to arrive at an overall architecture for CIEs. The core difference between a DODE and a CIE is that a DODE focuses on delivering domain knowledge, conceived of as relatively static and universal, while a CIE is built around forms of community memory, treated as constantly evolving and largely specific to a particular community of practice. Where DODEs relied heavily on a set of critic rules predefined as part of the domain knowledge, CIEs generalize the function of the critiquing mechanisms.

In a CIE it is still necessary to maintain some representation of the task as a basis for the software to take action. This is most naturally accomplished if work is done *within* the software environment. If communication about designs takes place within the system where the design is constructed, then annotations and email messages can be linked directly to the design elements they discuss. This reduces problems of deixis (comments referring to "that" object "over there" in a design). It allows related items to be linked together by automatic analysis mechanisms. In a rich community memory there may be many relationships of interest between new work artifacts and items in the information spaces. For instance, when a LAN manager debugs a network then links between network diagrams, topology designs, LAN diary entries, device tables or an interactive glossary of local terminology can be browsed to discover relevant information on demand.

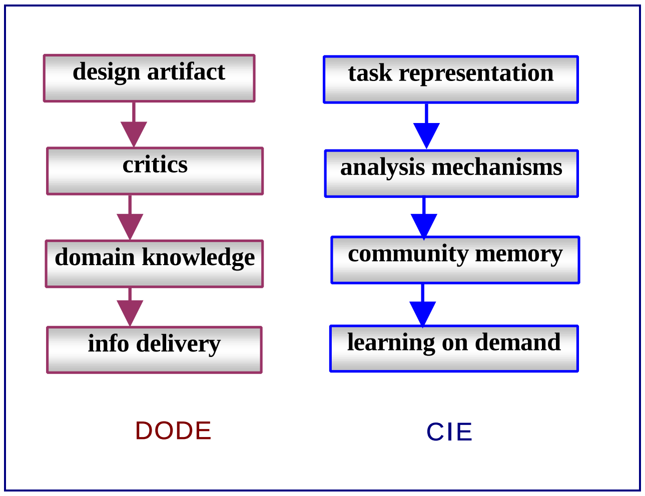


Figure 3. Generalization of DODE architecture (left) to a CIE structure (right).

The general problem for a CIE is to define *analysis mechanisms* that can bridge from the *task representation* to relevant *community memory* information items to support *learning on demand* (see Figure 3, right).

To take a very different example, suppose you are writing a paper within a software environment that includes a digital library of papers written by you and your colleagues. Then an *analysis mechanism* to support your learning might compare sentences or paragraphs in your draft (which functions as a *task representation*) to text from other papers and from email discussions (the *community memory*) to find excerpts of potential interest to deliver for your *learning*. We use latent semantic analysis [16] to mine our email repository [18] and are exploring similar uses of this mechanism to link task representations to textual information to support organizational learning.

The impetus for our extending DODEs into CIEs came partially from the advent of the World Wide Web. This technology facilitates the sharing and collaborative evolving of information and computer support within a community of practice—even within dispersed or virtual communities. In 1996/97 we prototyped WEBNET [35], a CIE for LAN management communities, as a web-based intranet. It includes a variety of communication media as well as community memory repositories and collaborative productivity tools (see Figure 4, left frame). Our work on WEBNET started with ProNet [32], a Mac-based DODE for LAN design, and gradually adapted it to the web.

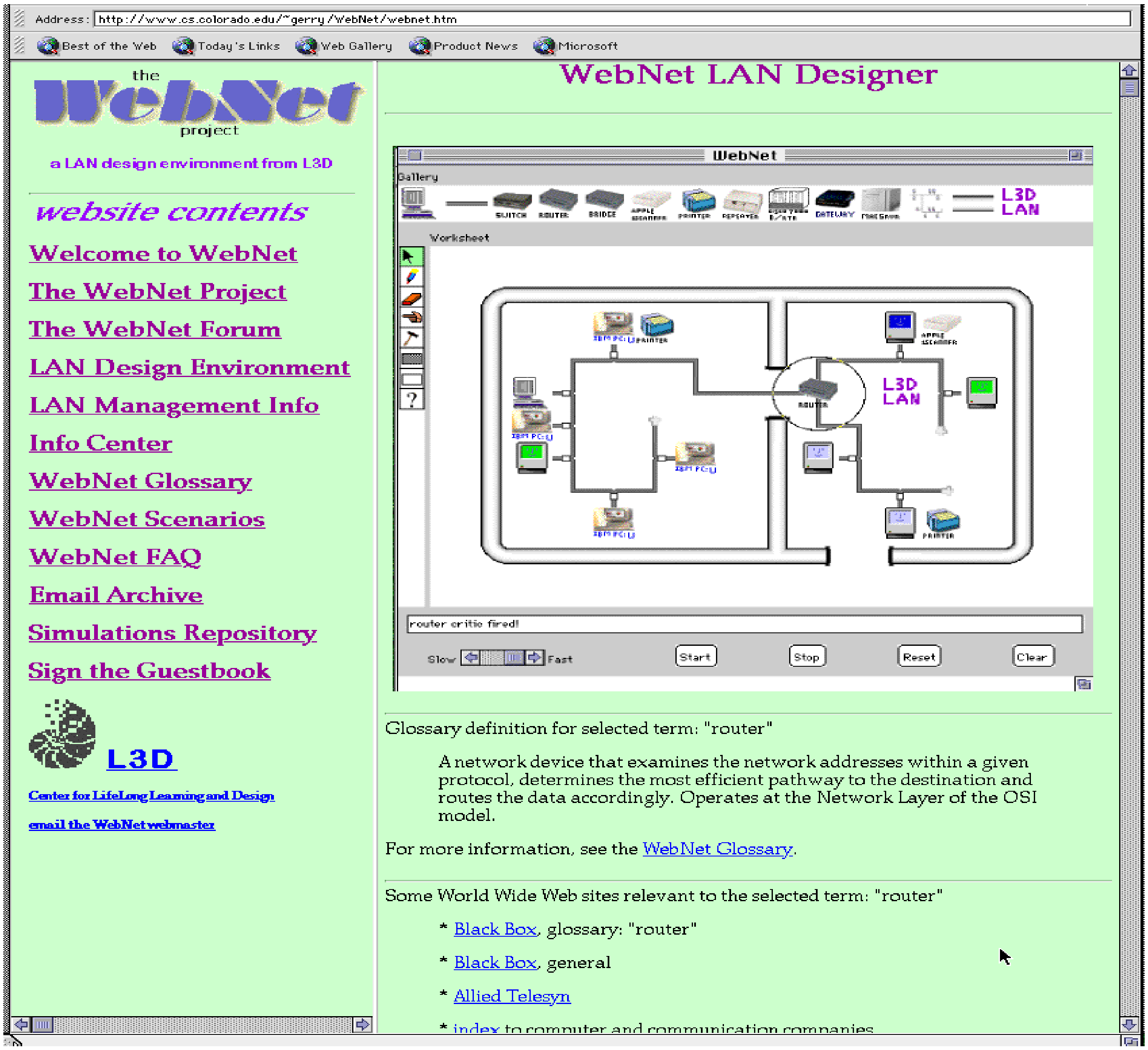
The web has the potential to support the interactivity needed for CIEs to maintain community memories. Dynamic web pages can be *interactive* in the sense that they accept user inputs through selection buttons and text entry forms. Unlike most forms on the web that only provide information (like product orders, customer preferences or user demographics) to a site webmaster, intranet feedback may be made immediately available to the user community that generated it.

The following WEBNET scenario includes examples of interactive community memories embedded in a set of communication and information delivery components. It demonstrates how intranet technology can be used by CIEs in which community members deposit knowledge as they acquire it so that other members can learn when they need to or want to and can communicate about the new knowledge.

SCENARIO OF A CIE IN USE

Delivering Web-based Information

Kay is a graduate student who works part-time to maintain



router critic fired!

Glossary definition for selected term: "router"

A network device that examines the network addresses within a given protocol, determines the most efficient pathway to the destination and routes the data accordingly. Operates at the Network Layer of the OSI model.

For more information, see the WebNet Glossary.

Some World Wide Web sites relevant to the selected term: "router"

* Black Box, glossary: "router"
* Black Box, general
* Allied Telesyn
* index to computer and communication companies

Figure 4. The WEBNET LAN design and simulation workspace (upper-right frame) and information delivered by a critic (lower-right frame). Note table of contents to the web site (left frame).

her department's LAN. The department has a budget to extend its network and has asked Kay to come up with a design. Kay brings up WEBNET in her web browser at http://www.cs.colorado.edu/~gerry/WebNet/webnet.htm.

She opens up the design of her department's current LAN in the LAN Design Environment, an AGENTSHEETS [27] simulation applet. Kay starts to add a new subnet. Noticing that there is no icon for an Iris graphics workstation in her palette, Kay selects the WEBNET menu item for the Simulations Repository web page. This opens a web site that contains simulation agents that other AGENTSHEETS users have programmed. WEBNET opens the repository to display agents that are appropriate for WEBNET simulations. Kay locates an agent that someone else has created with the behavior of an Iris workstation. She adds this to her palette and to her design.

When Kay runs the LAN simulation WEBNET proactively [32] inserts a router (see Figure 4, upper right), and informs Kay that a router is needed at the intersection of the two subnets. WEBNET displays some basic information about routers and suggests several web bookmarks with details about different routers from commercial vendors (see Figure 4, lower right). Here, WEBNET has signaled a breakdown in Kay's designing and provided easy access to sources of information for her to learn what she needs to know on demand. This information includes generic domain knowledge like definitions of technical terms, current equipment details like costs and community memory from related historical emails.

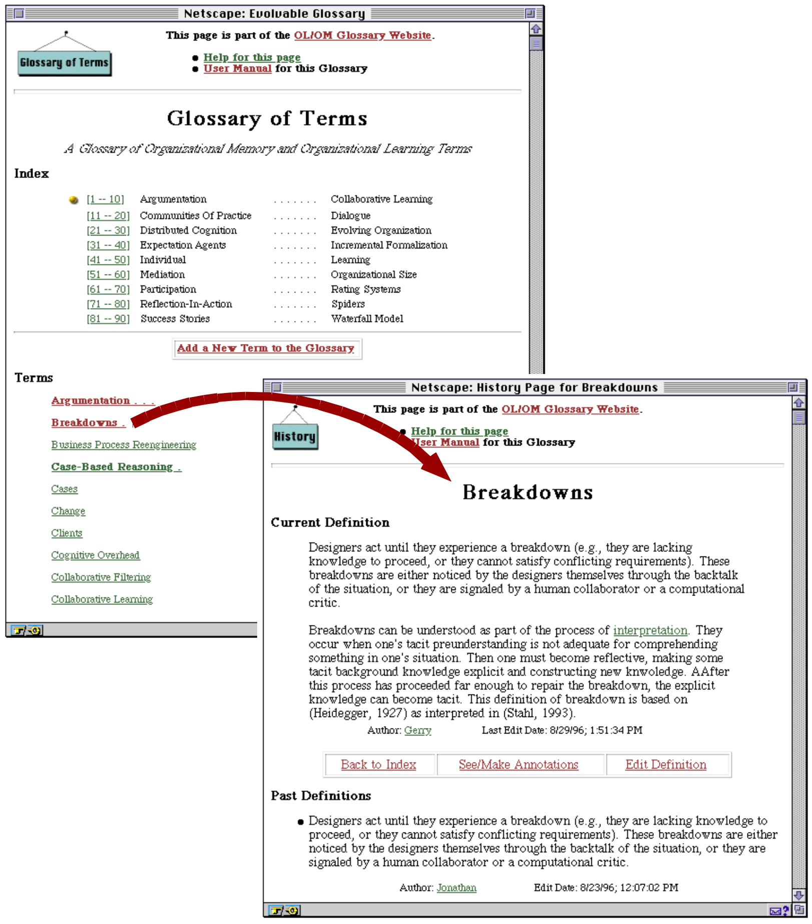
WEBNET points to several email messages from Kay's colleagues that discuss router issues and how they have been handled locally. The Email Archive includes all emails sent to Kay's LAN management workgroup in the past. Relevant emails are retrieved and ordered by the Email Archive software [18] based on their semantic relatedness to a query. In Kay's situation, WEBNET automatically generates a query describing the simulation context, particularly the need for a router. The repository can also be browsed, using a hierarchy of categories developed by the user community.

Kay reviews the email to find out which routers are preferred by her colleagues. Then she looks up the latest specs, options and costs on the web pages of router suppliers. Kay adds the router she wants to the simulation and re-runs the simulation to check it. She saves her new design in a catalog of local LAN layouts. Then she sends an email message to her co-workers telling them to take a look at the new design in WEBNET's catalog. She also asks Jay, her mentor at Network Services, to check her work.

Interactive and Evolving Knowledge

Jay studies Kay's design in his web browser. He realizes that the Iris computer that Kay has added is powerful enough to perform the routing function itself. He knows that this knowledge has to be added to the simulation in order to make this option obvious to novices like Kay when they work in the simulation. AGENTSHEETS includes an end-user programming language that allows Jay to reprogram the Iris workstation agent. To see how other people have programmed similar functionality, Jay finds a server agent on the Simulations Repository and looks at its program. He adapts it to modify the behavior of the Iris agent and stores this agent back on the repository. Then he redefines the router critic rule in the simulation. He also sends Kay an email describing the advantages of doing the routing in software on the Iris; WEBNET may make this email available to people in situations like Kay's in the future.

When he is finished, Jay tests his changes by going through the process that Kay followed. This time, the



This page is part of the OL/OM Glossary Website.

- Help for this page
- User Manual for this Glossary

Glossary of Terms

A Glossary of Organizational Memory and Organizational Learning Terms

Index

[1 -- 10]	Argumentation	. . . .	Collaborative Learning
[11 -- 20]	Communities Of Practice	. . . .	Dialogue
[21 -- 30]	Distributed Cognition	. . . .	Evolving Organization
[31 -- 40]	Expectation Agents	. . . .	Incremental Formalization
[41 -- 50]	Individual	. . . .	Learning
[51 -- 60]	Mediation	. . . .	Organizational Size
[61 -- 70]	Participation	. . . .	Rating Systems
[71 -- 80]	Reflection-In-Action	. . . .	Spiders
[81 -- 90]	Success Stories	. . . .	Waterfall Model

Add a New Term to the Glossary

Terms

Argumentation . . .
Breakdowns .
Business Process Reengineering
Case-Based Reasoning .
Cases
Change
Clients
Cognitive Overhead
Collaborative Filtering
Collaborative Learning

This page is part of the OL/OM Glossary Website.

- Help for this page
- User Manual for this Glossary

Breakdowns

Current Definition

Designers act until they experience a breakdown (e.g., they are lacking knowledge to proceed, or they cannot satisfy conflicting requirements). These breakdowns are either noticed by the designers themselves through the backtalk of the situation, or they are signaled by a human collaborator or a computational critic.

Breakdowns can be understood as part of the process of interpretation. They occur when one's tacit preunderstanding is not adequate for comprehending something in one's situation. Then one must become reflective, making some tacit background knowledge explicit and constructing new knwoledge. AAfter this process has proceeded far enough to repair the breakdown, the explicit knowledge can become tacit. This definition of breakdown is based on (Heidegger, 1927) as interpreted in (Stahl, 1993).

Author: Gerry Last Edit Date: 8/29/96; 1:51:34 PM

Back to Index | See/Make Annotations | Edit Definition

Past Definitions

- Designers act until they experience a breakdown (e.g., they are lacking knowledge to proceed, or they cannot satisfy conflicting requirements). These breakdowns are either noticed by the designers themselves through the backtalk of the situation, or they are signaled by a human collaborator or a computational critic.

Author: Jonathan Edit Date: 8/23/96; 12:07:02 PM

Figure 5. A view of the WEBNET Glossary. The current definition of a term is displayed. The history of definitions shows an earlier, modified definition. Note the ability to "See/Make Annotations" and to "Edit Definition."

definition of router supplied by WEBNET catches his eye. He realizes that this definition could also include knowledge about the option of performing routing in workstation software. The definitions that WEBNET provides are stored in an interactive glossary (see Figure 5). Jay goes to the WEBNET glossary entry for "router" and clicks on the "Edit Definition" button. He adds a sentence to the existing definition, noting that routing can sometimes be performed by server software. He saves this definition and then clicks on "Make Annotations". This lets him add a comment suggesting that readers look at the simulation he has just modified for an example of software routing. Other community members may add their own comments, expressing their views of the pros and cons of this approach. Any glossary user can quickly review the history of definitions and comments–as well as contribute their own thoughts.

Community Memory

It is now two years later. Kay has graduated and been replaced by Bea. The subnet that Kay had added crashed last night due to print queue problems. Bea uses the LAN Management Information component of WEBNET to trace back through the history of problems and changes leading up to the print queue problem.

The LAN Management Information component of WEBNET consists of four integrated information sources: a Trouble Queue of reported problems, a Host Table listing device configurations, a LAN Diary detailing chronological modifications to the LAN and a Technical Glossary defining local hardware names and aliases. These four sources are accessed through a common interface that provides for interactivity and linking of related items.

The particular problem that Bea is working on was submitted to her through the Trouble Queue; her solution will be added there to provide documentation. Bea starts her investigation with the Host Table, reviewing how the printer, routers and servers have been configured. This information includes links to LAN Diary entries dating back to Kay's work and providing the rationale for how decisions were made by the various people who managed the LAN. Bea also searches the Trouble Queue for incidents involving the print queue and related device configurations. Many of the relevant entries in the four sources are linked together, providing paths to guide Bea on an insightful path through the community history. After successfully debugging the problem using the community memory stored in WEBNET, Bea documents the solution by making entries and new cross links in the LAN Management Information sources.

In this scenario, Kay, Jay and Bea have used WebNet as a design, communication and memory system to support both their immediate tasks and the future work of their community.

CONCLUSION

The CIE concept arose from our work on the WEBNET prototype and our investigations of the needs of LAN management communities. WEBNET began as a port of a DODE for LAN design to the web. In the process, we came to recognize the importance of supporting evolving community memory with interactive intranet technology. The DODE focus on domain-oriented simulations, critics and design rationale had to be extended with more communication and information delivery mechanisms.

Technical domains are too complex, fast changing and locally variable to expect a vendor of DODEs like NETSUITE to maintain adequate knowledge repositories. Local information is even harder than generic domain information for outside knowledge engineers to compile, being largely tacit expertise of community old-timers. So it is up to community members to maintain information collaboratively in a distributed fashion. But busy people cannot be burdened with massive data entry tasks whose payoff seems remote. In our current research, we are exploring the following approaches to this problem:

- Allow community members to build knowledge by commenting naturally on information as they encounter it in their regular work. Most information (like glossary definitions) should be interactive, allowing for immediate annotation and revision whenever appropriate.
- Embed communication about artifacts within the same system as work on the artifact. Then the messages can be archived and associated with the artifact automatically.
- Allow community members to link and reorganize information in order to build and update useful structuring of the information space. Support these efforts with automation where possible.
- Help community members to personalize information delivery with adaptable features. Enhance this with automatic adaptation of the system to a member's preferences and needs.

A CIE should be a high-functionality software environment in which people work, communicate and learn collaboratively:

- It should incorporate tools for engaging in the work practices of the group.
- It should support multiple modes of communication, such as Internet chat, email, threaded discussions, ubiquitous annotation.
- It should deliver timely, relevant information to support lifelong learning in collaborative settings.

Community memory may be most effective when embedded in a CIE. Emerging intranet technology provides the technological basis for effective systems built around community memories for learning in communities of practice. However, features, techniques and practices to realize this potential are just beginning to be investigated. While some of our early ideas for DODEs have matured into current best practices, there are still many open research issues surrounding how to realize the potential of CIEs for supporting collaborative work within specific communities of practice.

ACKNOWLEDGMENTS

We would like to thank the other members of the Center for LifeLong Learning and Design, particularly the Organizational Memory and Organizational Learning group, including Jay Smith, Scott Berkebile, Sam Stoller, Jim Masson and Tim Ohara who worked on the WEBNET system. Our knowledge of LAN design benefited from our domain ethnographers John Rieman and Ken Anderson and local informants Kyle Kucson and Evi Nemeth. The work reported here was supported in part by ARPA grant N66001-94-C-6038 and NSF grant IRI-9711951. NETSUITE ADVANCED PROFESSIONAL DESIGN is a trademark of NetSuite.

REFERENCES

1. Ackerman, M. S. (1994) Augmenting the organizational memory: A field study of Answer Garden. *Proceedings of CSCW '94,* ACM Press, 243-252.
2. Boland, R. J. Jr. & Tenkasi, R. V. (1995) Perspective making and perspective taking in communities of knowing. *Organization Science*. *6*, 4, 350-372.
3. Bourdieu, P. (1972) *Esquisse d'une theorie de la pratique*. Switzerland: Librairie Droz, S. A.
4. Brown, J. S. & Duguid, P. (1991) Organizational learning and communities of practice: Toward a unified view of working, learning, and innovation. *Organization Science*. *2*, 1, 40-57.
5. Donald, M. (1991) *Origins of the Modern Mind*. Cambridge, MA: Harvard University Press.
6. Fischer, G. (1996) Making learning a part of life: Beyond the "gift wrapping" approach of technology. In: Alheit & Kammler (Eds.) *Lifelong Learning and its Impact on Social and Regional Development*. Bremen, Germany: Donat Verlag. 435-462. Available at http://www.cs.colorado.edu/~l3d/presentations/gf-wlf/.
7. Fischer, G. (1994) Domain-oriented design environments. *Automated Software Engineering*. *1*, 2, 177-203.
8. Fischer, G. (1994) Turning breakdowns into opportunities for creativity. *Knowledge-Based Systems*. *7*, 4, 221-232.
9. Fischer, G. (1991) Supporting learning on demand with design environments. *International Conference on the Learning Sciences*. 165-172.
10. Fischer, G. (1989) Creativity enhancing design environments. *Proceedings of the International Conference on Modeling Creativity and Knowledge-Based Creative Design*. Heron Island, Australia. 127-132.
11. Fischer, G. (1987) A critic for Lisp. *IJCAI*. 177-184.
12. Fischer, G., McCall, R., Ostwald, J., Reeves, B. & Shipman F. (1996) Seeding, evolutionary growth and reseeding: The incremental development of collaborative design environments. In: Olson, G., Malone T. & Smith, J. (Eds.): *Coordination Theory and Collaboration Technology*.
13. Fischer, G., Nakakoji, K., Ostwald, J., Stahl, G. & Sumner, T. (1993) Embedding critics in design environments. *The Knowledge Engineering Review*. *8*, 4, 285-307.
14. Fischer, G., Grudin, J., Lemke, A., McCall, R., Ostwald, J., Reeves, B., Shipman, F. (1992) Supporting indirect, collaborative design with integrated knowledge-based design environments. *HCI*. *7*, 3, 281-314
15. Giddens, A. (1984) *The Constitution of Society*. Berkeley: University of California Press.
16. Landauer, T. K. & Dumais, S. T. (1997): A solution to Plato's problem: The latent semantic analysis theory of the acquisition, induction, and representation of knowledge. *Psychological Review*. *104*, 211-240.
17. Lave, J. & Wenger, E. (1991) *Situated Learning: Legitimate Peripheral Participation*. New York: Cambridge University Press.
18. Lindstaedt, S. (1997) Towards organizational learning: Growing group memories in the workplace. *Proceedings of CHI '96*. Doctoral Consortium. Vancouver, British Columbia, Canada.
19. Marx, K. (1867) *Das Kapital: Kritik der politischen Oekonomie*. Erster Band. Hamburg: Verlag von Otto Meissner.
20. Nardi, B. (1993) *A Small Matter of Programming*. Cambridge, MA: MIT Press.
21. NetSuite Advanced Professional Design home page. Available at http://www.netsuite.com/products/docs/napd.htm.
22. Norman, D. (1993) *Things That Make Us Smart*. Reading, MA: Addison-Wesley.
23. Organizational Memory and Organizational Learning project home page. Available at http://www.cs.colorado.edu/~l3d/omol.
24. Orr, J. (1996) *Talking about Machines*. Ithaca, NY: Cornell University Press.
25. Ostwald, J. (1996*) Knowledge Construction in Software Development: The Evolving Artifact Approach*. Un-

published Ph.D. Dissertation. Department of Computer Science. University of Colorado.

26. Polanyi, M. (1962) *Personal Knowledge*. London: Routledge & Kegan Paul.

27. Repenning, A. (1994) Programming substrates to create interactive learning environments. *Journal of Interactive Learning Environments. 4*, 1, 45-74.

28. Rittel, H. & Webber, M. (1984) Planning problems are wicked problems. In Cross, N., *Developments in Design Methodology*. New York: Wiley. 135-144.

29. Schön, D. (1983) *The Reflective Practitioner*. New York: Basic Books.

30. Stahl, G. (1993) Supporting situated interpretation. *Proceedings of the Cognitive Science Society*. 965-970. Available at http://www.cs.colorado.edu/~gerry/HomePage/Publications/CogSci/CogSci.html.

31. Stahl, G. (1993) *Interpretation in Design: The Problem of Tacit and Explicit Understanding in Computer Support of Cooperative Design*. Unpublished Ph.D. Dissertation. Department of Computer Science. University of Colorado.

32. Sullivan, J. (1994) *A Proactive Computational Approach for Learning While Working*. Unpublished Ph.D. Dissertation. Department of Computer Science. University of Colorado.

33. Sumner, T. (1995) *Designers and their Tools: Computer Support for Domain Construction*. Unpublished Ph.D. Dissertation. Department of Computer Science. University of Colorado.

34. Tomasello, M., Kruger, A. C. & Ratner, H. (1993) Cultural learning. *Behavioral and Brain Sciences*. 495-552.

35. WebNet home page. Available at http://www.cs.colorado.edu/~gerry/WebNet.

36. Zuboff, S. (1988) *In the Age of the Smart Machine*. New York: Basic Books.