Collaborative Information Environments for Innovative Communities of Practice

Gerry Stahl
Center for LifeLong Learning and Design
University of Colorado at Boulder, USA

Abstract

In the information age, lifelong learning and collaboration are essential aspects of most innovative work. Fortunately, the computer technology which drives the information explosion also has the potential to help individuals and teams to learn much of what they need to know on demand. In particular, computer-based systems on the Internet can be designed to capture knowledge as it is generated within a community of practice and to deliver relevant knowledge when it is useful. 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, these *domain-oriented design environments* (DODEs) must be extended to become *collaborative information environments* (CIEs), capable of providing effective community memories for managing information and learning within constantly evolving collaborative contexts.

1 Computer Support for Individual Innovation

1.1 The Need for LifeLong Learning for Innovation

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.
- There is too much knowledge, even within specific subject areas, for anyone to master it all in advance or on one's own.
- The knowledge in many domains evolves rapidly and often depends upon the context of one's task situation, including one's support community.
- 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.
- People's careers and self-directed interests require various new forms of learning at different stages as their roles in communities change.
- Learning — especially collaborative learning — has become a new form of labor, an integral component of work and organizations.

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 organizational 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.

1.2 Domain-Oriented Design Environments

Many innovative work tasks can be conceived of as *design* processes: elaborating a new idea, planning a presentation, balancing conflicting proposals or writing a visionary report, for example. 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 [Schön 83]. Thereby, designing entails learning.

For the past decade, researchers at the University of Colorado have explored the creation of DODEs to support workers as designers. These systems are *domain-oriented*: they incorporate knowledge specific to the work domain. They are able

to recognize when a breakdown in understanding has occurred and can respond to it with appropriate information [Fischer 89].

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 the artifact being constructed. 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 (the primary example in this paper) allows a designer to construct a network design by arranging items from a palette 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* [Fischer et al. 93a]. 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 instance, a LAN DODE might notice that the length of a cable in a design 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 [Fischer et al. 91].

1.3 A Commercial Product

Many of the ideas in our DODEs are now appearing in commercial products, independently of our efforts. In particular, there are environments for designing LANs. As an example, consider NETSUITE [NetSuite 97], a highly rated system that illustrates current best practices in LAN design support. 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 that can be placed in the construction area 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

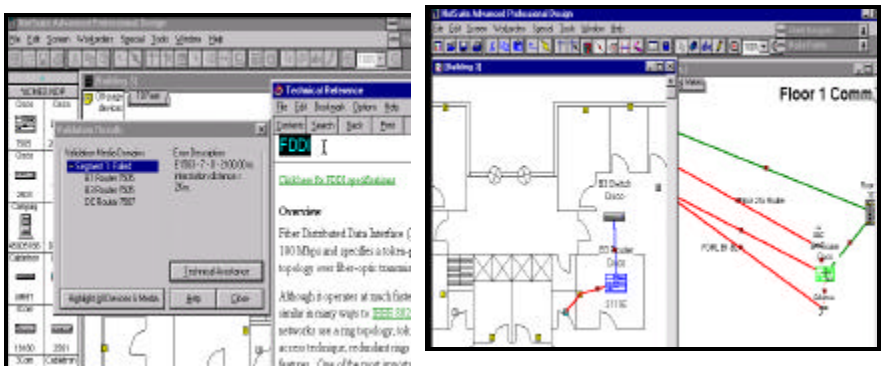

Figure 1. Two views of NETSUITE. In the left view, the system has noted that a cable length specification for a FDDI network has been exceeded in the design and the system has delivered information about the specification and affected devices. In the right view, parts of the network viewed in physical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

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.

1.4 The Need to Go Further

Based on our understanding of organizational learning 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, too constantly changing and too dependent upon local circumstances. The next generation of commercial DODEs will have to support *extensibility* by end-users 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 managers involves 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 [Lave & Wenger 91]. 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 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 [Fischer et al. 96]. DODEs must be extended to support communities of practice, not just isolated designers.

2 Supporting Communities of Practice

2.1 Communities of Practice

All work within a division of labor is social [Marx 1867]. The job that one person performs is also performed similarly by others and relies upon vast social networks. That is, work is defined by *social practices* that are propagated through socialization, apprenticeship, training, schooling, and culture [Giddens 84; Bourdieu 72], as well as by explicit standards. Often, work is performed by cooperating teams that form *communities of practice* within or across organizations [Brown & Duguid 91].

For instance, interviews we conducted 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 [Orr 90]. Support for LAN managers must provide 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 dynamic web sites, it is now

possible to support distributed communities and also to maintain interactive and evolving information about local circumstances and group history.

2.2 Digital 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 [Donald 91; Norman 93]. 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 work practices. New computer-based organizational 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. They make organizational learning possible within the group.

1.3 The Process of Organizational Learning

The ability of designers to proceed based on their tacit existing expertise [Polanyi 62] periodically breaks down and they have to rebuild their understanding of the situation through explicit reflection [Schön 83]. 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) [Stahl 93a].

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 personal reflection

and collaborative articulation of shared perspectives makes innovation possible [Boland et al. 95; Tomasello et al. 93].

Organizational learning can be supported by computer-based systems of organizational memory 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) [Stahl 93b]. DODEs provide computer support for design by individuals. They need to be extended to collaborative information environments (CIEs) to support organizational learning in communities of practice.

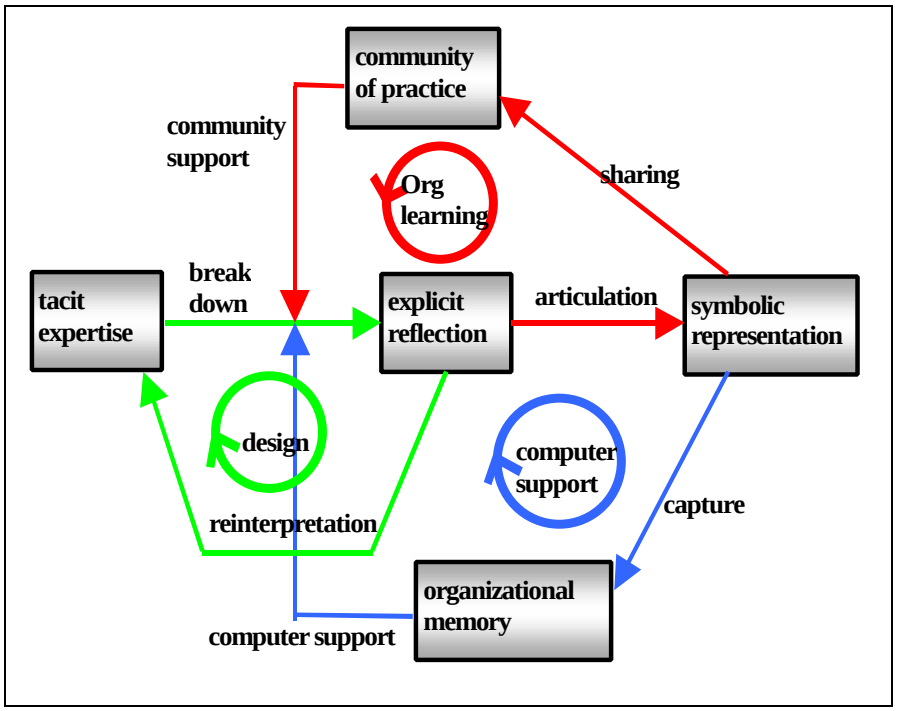

Figure 2. Cycles of design, computer support and organizational learning.

1.4 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 [Fischer et al. 93b]. 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 manner.

Systems 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 organizational memories. 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. For instance, 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"). It also allows related items to be linked together automatically. In a rich information space there may be many relationships of interest between new work artifacts and items in the organizational memory. For instance, when a LAN manager debugs a network, links between network diagrams, topology designs, LAN diary entries, device tables and an interactive glossary of local terminology can be browsed to discover relevant information.

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

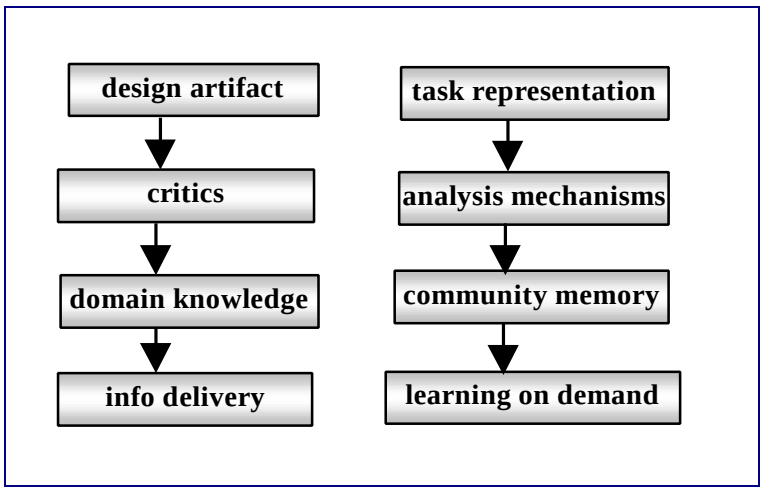

Figure 3. Generalization of the DODE architecture (left) to a CIE (right).

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 [Landauer & Dumais 97] to mine our email repository [Lindstaedt 97] and are exploring similar uses of this mechanism to link task representations to textual information to support organizational learning. Other retrieval mechanisms might be appropriate for mining catalogs of software agents or components, design elements and other sorts of organizational memories.

1.5 Integrative Systems for Community Memory

Effective community memory relies on integration. Tools for representing design artifacts and other work tasks must be related to rich repositories of information that can be brought to bear when needed. Communication about artifacts under development should be embedded in the artifact so they retain their context of significance and their association with each other. Finally, members of the community of practice must be integrated with each other in ways that allow something one member learned in the past to be delivered to other members when they need it in the future. One model for such integration — on an individual level — is the human brain, which stores a wealth of memories over a lifetime of experience, thought and learning in a highly inter-related associative network that permits effective recall based on relevance.

A traditional way to integrate information in a computer system is with a relational database. This allows associations to be established among arbitrary data. It also provides mechanisms like SQL queries to retrieve information based on specifications in a rather comprehensive language. Integrating all the information of a design environment in a unified database makes it possible to build bridges from the current task representation to any other information. Of course, object-oriented or hybrid databases and distributed systems that integrate data on multiple computers can provide the same advantages. Nor does an underlying query language like SQL have to be exposed to users; front-end interfaces can be much more graphical and domain-oriented.

Communities must also be integrated. The Internet provides a convenient technology for integrating the members of a community of practice, even if they are physically dispersed or do not share a homogeneous computer platform. In particular, *intranets* are web sites designed for communication within a specific community rather than world-wide. WEBNET, for instance, is an intranet that we prototyped for LAN management communities. It includes a variety of communication media as well as community memory repositories and collaborative productivity tools (see Figure 4, left frame).

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 Internet that only provide information (like product orders, customer preferences or user demographics) to the webmaster, intranet feedback may be made immediately available to the user community that generated it. For instance, the following scenario includes an interactive glossary. When someone modifies a glossary definition the new definition is displayed to anyone looking at the glossary. Community members can readily comment on the definitions or change them. The history of the changes and comments made by the community is shared by the group. In this way, intranet technology can be used to build systems that are 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 it. Let us imagine how a CIE like WEBNET might be used in a concrete scenario.

3 Scenario of a Collaborative Information Environment in Use

3.1 Critiquing and Information Delivery

Kay is a graduate student who works part-time to maintain 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 [Reppening 94] 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 a simulation 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 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 [Lindstaedt 97] 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.

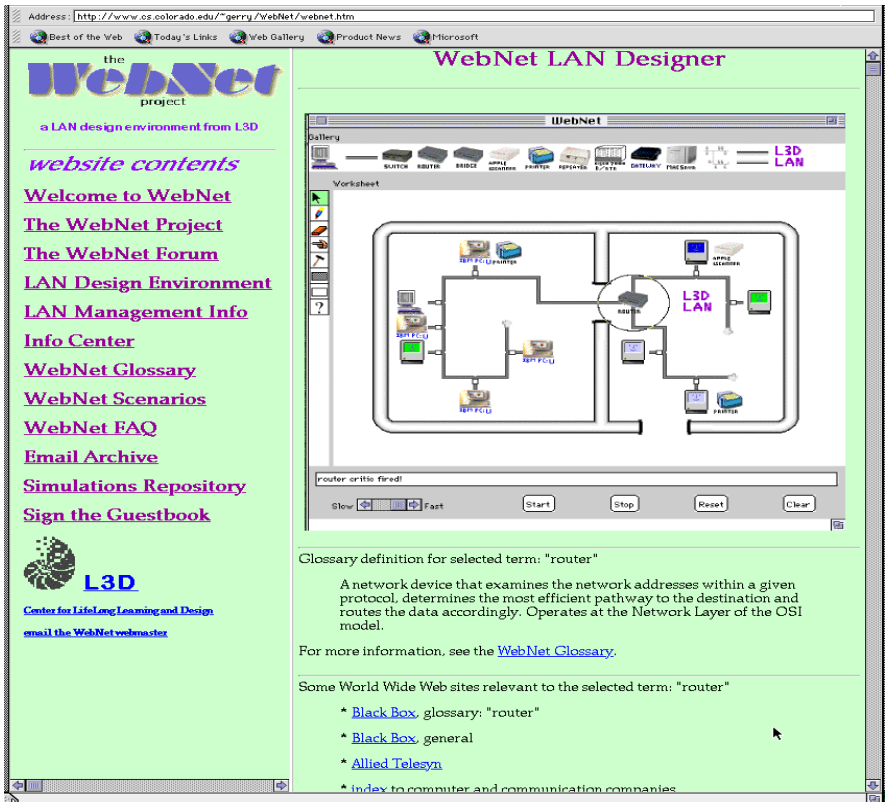

Figure 4. The WEBNET LAN design and simulation workspace (upper-right frame) and information delivered by a critic (lower-right frame). Note table of contents to the web site (left frame).

3.2 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 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. 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.

3.3 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 Info component of WEBNET to trace back through a series of email trouble reports and entries in LAN diaries. After successfully debugging the problem using the community memory stored in WEBNET, Bea documents the solution by making an entry in the technical glossary as well as noting detailed changes in the LAN diary.

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.

4 Vision of Support for Innovative Communities

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 rich repositories of domain knowledge. 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: a community memory must be implemented as a CIE.

But busy people cannot be burdened with massive data entry tasks whose payoff seems remote. The capturing of information in a CIE must be incorporated into the social practices of the community and it must be computationally supported to reduce the burden and maximize the benefits. In our current research, we are exploring the following approaches to this problem:

- Allow people to build knowledge by commenting naturally on information as they encounter it in their regular work. All information (like glossary definitions) should be interactive, allowing for immediate annotation and revision (subject, of course, to security, privacy and authority issues).

- 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 user's preferences and needs.

A CIE should be a high-functionality software environment in which people work, communicate and learn collaboratively. By serving as an environment for a community‘s activities, the CIE can capture work artifacts and work rationale in an organizational memory that informs innovation:

- 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, non-intrusive information to support lifelong learning.

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 innovation within specific communities of practice.

5 Acknowledgments

This research was a collaboration of the author with Gerhard Fischer and Jonathan Ostwald. We would like to thank the other members of the Center for LifeLong Learning and Design, particularly the Organizational Memory 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 investigators John Rieman and Ken Anderson and local informants Kyle Kucson and Evi Nemeth. The work reported here was supported in part by grants from ARPA N66001-94-C-6038 and NSF IRI-9711951. NETSUITE ADVANCED PROFESSIONAL DESIGN is a trademark of NetSuite.

6 References

[Boland et al. 95] Boland, R. J. Jr. & Tenkasi, R. V. (1995): Perspective Making and Perspective Taking in Communities of Knowing. *Organization Science*. Vol. 6, No. 4, Pp. 350-372.

[Bourdieu 72] Bourdieu, P. (1972): *Esquisse d'une theorie de la pratique*. Switzerland: Librairie Droz, S. A.

[Brown & Duguid 91] Brown, J. S. & Duguid, P. (1991): Organizational Learning and Communities of Practice: Toward a Unified View of Working, Learning, and Innovation. *Organization Science*. Vol. 2. No. 1. Pp. 40-57.

[Donald 91] Donald, M. (1991): *Origins of the Modern Mind*. Cambridge, MA: Harvard University Press.

[Fischer 89] Fischer, G. (1989): Creativity Enhancing Design Environments. Proceedings of the International Conference on Modeling Creativity and Knowledge-Based Creative Design. Heron Island, Australia. Pp. 127-132.

[Fischer et al. 96] 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*.

[Fischer et al. 93a] Fischer, G.; Nakakoji, K.; Ostwald, J.; Stahl, G. & Sumner, T. (1993): Embedding Computer-Based Critics in the Contexts of Design. *Proceedings of InterCHI '93*. Amsterdam. Pp. 157-164.

[Fischer et al. 93b] Fischer, G.; Nakakoji, K.; Ostwald, J.; Stahl, G. & Sumner, T. (1993): Embedding Critics in Design Environments. *The Knowledge Engineering Review*. Vol. 8. No. 4. Pp. 285-307. In: Maybury, M. & Wahlster, W. (1998): *Readings in Intelligent User Interfaces*. San Francisco, CA: Morgan-Kaufmann. Pp. 537-561.

[Fischer et al. 91] Fischer, G.; Lemke, A.; McCall, R. & Morch, A. (1991): Making Argumentation Serve Design. *Human-Computer Interaction*. Vol 6, Nos, 3 & 4. Pp. 393-419.

[Giddens 84] Giddens, A. (1984): *The Constitution of Society*. Berkeley: University of California Press.

[Landauer & Dumais 97] 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*. No. 104. Pp. 211-240.

[Lave & Wenger 91] Lave, J. & Wenger, E. (1991): Situated Learning: Legitimate Peripheral Participation. New York: Cambridge University Press.

[Lindstaedt 97] Lindstaedt, S. (1997): Towards Organizational Learning: Growing Group Memories in the Workplace. *Proceedings of CHI '96*. Doctoral Consortium. Vancouver, British Columbia, Canada.

[Marx 1867] Marx, K. (1867): *Das Kapital: Kritik der politischen Ökonomie*. Erster Band. Hamburg: Verlag von Otto Meissner.

[NetSuite 97] NetSuite Advanced Professional Design home page. Available at http://www.netsuite.com/products/docs/napd.htm.

[Norman 93] Norman, D. (1993): *Things That Make Us Smart*. Reading, MA: Addison-Wesley.

[Orr 90] Orr, J. (1990): Sharing Knowledge, Celebrating Identity: War Stories and Community Memory in a Service Culture. In: Middleton, D. S. & Edwards, D. (Eds.): *Collective Remembering: Memory in Society*. Beverly Hills, CA: Sage Publications.

[Polanyi 62] Polanyi, M. (1962): *Personal Knowledge*. London: Routledge & Kegan Paul.

[Reppening 94] Repenning, A. (1994): Programming Substrates to Create Interactive Learning Environments. *J. of Interactive Learning Environments. 4*, 1, Pp. 45-74.

[Schön 83] Schön, D. (1983): *The Reflective Practitioner*. New York: Basic Books.

[Stahl 93a] Stahl, G. (1993) Supporting Situated Interpretation. *Proceedings of the Cognitive Science Society*. Boulder, CO. Pp. 965-970.

[Stahl 93b] 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.

[Tomasello et al. 93] Tomasello, M.; Kruger, A. C. & Ratner, H. (1993): Cultural Learning. *Behavioral and Brain Sciences*. Pp. 495-552.