Minimalist Design for Informal Learning in Community Computing

Mary Beth Rosson, John M. Carroll

Center for Human-Computer Interaction, Pennsylvania State University

mrosson@psu.edu, jcarroll@ist.psu.edu

Abstract. We discuss the role and characteristics of informal learning in a community computing context. We argue that minimalist design can be adapted to the needs of community computing, and that its principles can be used to envision and develop community activities and technologies that promote active learning. We illustrate these ideas with several community computing projects that exemplify how to embed learning in meaningful activities, enable learners to make progress quickly, promote thinking and inference, evoke and leverage prior knowledge, and support error recognition and recovery. We conclude with a discussion of how minimalism might be used more broadly to guide the design of community computing systems and activities.

Informal Learning in Community Computing

Community computing refers to the use of networking, software, and activities to support community interactions. Classic examples include the Cleveland Free Net (facilitating dissemination of public health information; Beamish, 1995), and Montana's Big Sky Telegraph (improving teachers' access to library resources in rural Montana; Uncapher, 1999). Community computing is distinctive in that participants are neighbors in the traditional sense; they live in physical proximity and share physical, economic and social resources. In contrast to Internet communities, the information character of community computing is primarily local—description, news, and events pertaining to clubs and churches, public schools, municipal government, voluntary associations, retail businesses, regional economic development and social services.

An important facet of community computing is *learning*. Learning may occur as an incidental outcome but is an inherent consequence of any creative human activity; people learn even when they "merely" pursue familiar interests and concerns in new ways. The Cleveland and Montana communities did not simply accomplish community goals more efficiently; they evolved a local Internet culture of new skills and practices. They learned how to use Internet technologies to accomplish new community functions. In Cleveland community health information dissemination evolved into community health discussions through which the hospital staff learned about the needs of their customers. In Montana, people accessed online libraries for existing information; the community created novel coding schemes to digitize Native American cultural artifacts, enabling cultural dissemination back *into* the libraries.

Communities support technology learning in many ways, including just-in-time training classes offered through adult learning centers, community colleges, and so on. In this paper we focus on *informal learning* (IL)*, working from McGiveney's (1998) broad characterization: informal learning occurs outside of intentional learning environments such as classrooms; arises through people's activities and interests; and even when provided in response to perceived needs is conveyed in a flexible and informal fashion. In the Cleveland and Montana examples, although there may have been an explicit effort to learn about community members' needs on the one hand, and coding schemes, on the other, the shared learning was emergent and entirely situated in the ongoing activities and goals of the community.

Community computing provides an especially rich—and challenging—context for analyzing and supporting informal learning processes (Figure 1). Community groups are often structured and managed in an *ad hoc* fashion. Goals and activities may vary dynamically as a function of membership, current leadership, and resources. Although this adds to the complexity of community group interaction, it also raises the opportunity for many different individuals to "step up" to the needs of their groups.

Within these dynamic group structures, people enact multiple roles, relating to other members as parents, business colleagues, volunteers, and so on. Community members who *bridge* multiple groups have the potential for great impact on the community, in that they tend to be highly educated and more involved in civic concerns (Kavanaugh et al., 2003; Kavanaugh et al., 2005). Cross-group membership also has implications for informal learning, as people are able to see and act on the possibilities for cross-fertilization of ideas, methods, and resources (Putnam, 2000).

The members of community groups vary tremendously in what they bring to shared efforts: in business settings, people collaborate because they have a "job" to do, but in the discretionary context of community efforts, motivation is often intrinsic but varies considerably across individuals. The diverse motivations

interact with resource limitations such as time available, making it very difficult to develop and implement targeted learning opportunities. Also, because part of the shared capital in a community is a physical place, a variety of concrete experiential outcomes and rewards become more possible and common. Belonging to any community of practice implies a bond among members, but if we are managing the earth, this bond is inherently multifaceted and will lead to many levels and modes of interaction.

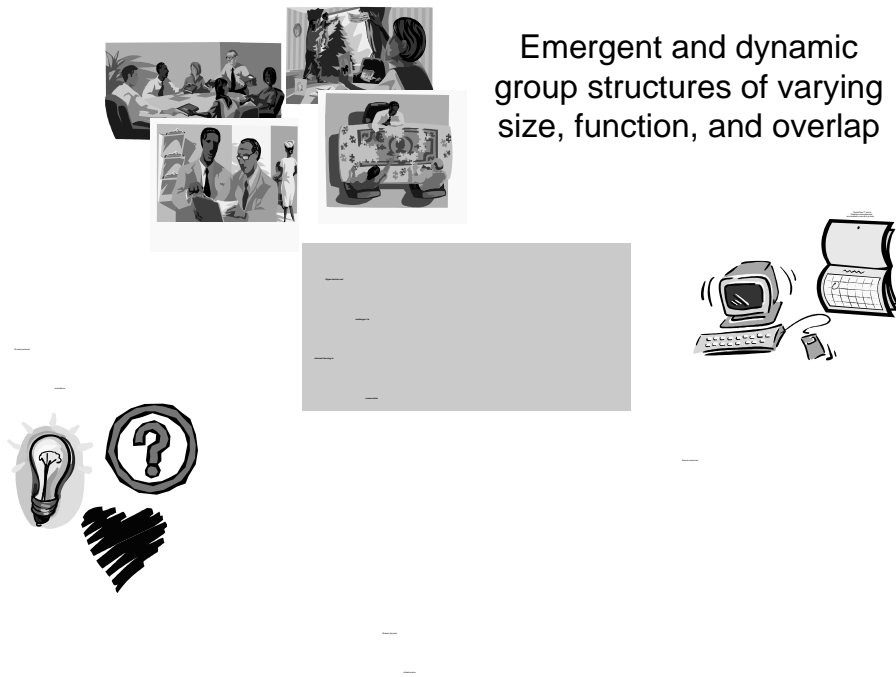Emergent and dynamic group structures of varying size, function, and overlap

Figure 1. Opportunities and challenges for learning in community computing

The unpredictability and complexity—but also the richness of motivations, backgrounds, and roles—of community computing highlight a need for informal, contextualized learning processes that leverage community members' goals, skills, time, technology, and other resources *if* and *when* they are available. In the balance of this paper we argue that minimalist design provides a guiding framework for initiating and sustaining a broad-based learning culture that accommodates the just-in-time context of community computing. We first briefly review the principles of minimalism and suggest how they can be generalized to address community computing contexts. We illustrate our arguments by reflecting on several of our own community computing projects, concluding with ideas for future analysis and discussion.

Minimalism as a Model for Informal Learning

Minimalism is an instructional design framework intended to address the needs of computer users as *active learners* (Carroll, 1990). Minimalism was a response to empirical studies of novice and experienced computer users as they attempted new tasks. Active learners often reject the prescribed goals and activities of training materials, instead developing and pursuing their own goals. They explore features in pursuit of these goals, working from whatever prior knowledge they possess, whether it is useful or not. When they make mistakes, active users try to understand what has happened, often learning something in the process. A key insight behind minimalism is that learning resources and tools should not protect users from these active learning tendencies but rather should *design for* the tendencies, thereby anticipating and supporting learning by doing.

Minimalist design can be summarized through five basic principles, each emphasizing an important characteristic of active learning that can be encouraged and reinforced through appropriate instructional design:

- *Embed learning in meaningful activities.* People are motivated by real tasks in the real world, not by artificial exercises that introduce new content bit by bit. Designers should offer learning tasks that connect with learners' real world goals and concerns but still achievable by novices.
- *Allow learners to get started quickly.* Learners want to take action and enjoy evidence of progress from the start. Minimalist designs should make it possible for learners to take concrete actions right away, rather than expecting preliminary effort such as reading or planning. Minimalist designs should coordinate presentation of information with opportunities for action to every extent possible (van der Meij & Carroll, 1995).
- *Rely on learners to think and to improvise.* When instructions tell learners what to do step by step, active users often respond by skipping around, in essence forcing themselves to think and improvise, but often ending up in error tangles. Designers instead should construct open-ended materials requiring inference while also offering tips for successful action.
- *Evoke and support the use of prior knowledge*. Learners understand new material with respect to what they already know, using analogies and generalization to make sense of new concepts. Designers should analyze learners' prior knowledge, using metaphors to promote learning transfer.
- *Prevent, mitigate and leverage errors.* Nuisance errors (like mis-typing a URL) should be anticipated and prevented when possible. But many errors are an opportunity for further learning when active users are considering and testing solutions. Designers should provide feedback that makes it clear when users are in an error state and offer tips that guide recovery.

Minimalist instruction was developed in response to human tendencies toward active learning. Because it emphasizes learning by doing, minimalism addresses

many of the characteristics of informal learning, for instance the connection of learning to real world activities and reliance on the learner to initiate and manage the reflection and learning process (Dewey, 1933) Minimalism views learners as motivated by intrinsic goals and learning as an ongoing process integrated with everyday experience. As such it is well suited to the needs of informal learning in community computing.

Minimalism in a Community Computing Context

Minimalism assumes that learners are situated in real world activities and that they use this real world context to make sense of new experiences with computing; however most work in minimalism has focused on sense-making by individuals (Carroll, 1998). To consider the role of minimalism for community computing, we must first consider how its principles can be extended or adapted to the problems of technology learning in a community context (Table 1).

Minimalist principle	Adaptation for community computing
Embed learning in meaningful activities	Embed learning in a rich and multi-faceted activity, supporting synthesis and compromise
Allow learners to get started quickly	Provide varied options for rapid results suitable for varied backgrounds, including social scaffolding
Rely on learners to think and to improvise	Design activities that require cooperation and have unpredictable emergent properties; provoke bridging, perspective-switching, and mutual adaptation
Evoke and support the use of prior knowledge	Encourage and support bridging among groups; make the highly distributed and evolving knowledge of the group visible and organized by familiar structures
Prevent, mitigate, and leverage errors	Make it safe and possible for members to monitor and contribute to others' work as well as their own; anticipate, support conflict recognition and resolution

Table 1. Adapting minimalism to a community computing context

One contrast between the body of work on minimalism and the context of community computing is that community learning is a shared and distributed process; it takes place at the level of groups rather than individuals. The learning that takes place is a collective achievement across a potentially diverse set of individuals with complementary or even competing personal motivations (Bertelsen & Bødker, 2003; Engeström, 1987). From a minimalist perspective, the important implication is that meaningfulness will be multi-faceted, with different stakeholders participating under different conceptions of what and how things are

happening. Imagine an activity in which volunteers make home technology visits to house-bound elders: the volunteers understand the activity as technology outreach, but for the elders the meaning may lie in the social interaction. The meaningfulness of the overall activity lies in the mutual recognition and synthesis of individuals' goals and motivations (Kuuti & Arvonen, 1982).

With respect to helping learners make progress quickly, a community context again suggests a more expansive view. Minimalist designs rely on tools or instructions that reduce preliminaries, focus learners' attention on just the right information, and offer a protected path to a meaningful goal (Carroll & Carrithers, 1984; Carroll & Kay, 1985; Rosson, Carroll & Bellamy, 1990). Such training techniques would surely be useful in community computing, but because of members' diverse background designers must offer multiple entry points into an activity, enabling the most novice members to participate when time is available, yet allowing sophisticated members to make progress at more challenging levels. By leveraging the indigenous knowledge and social structure of a community, designers can create social scaffolding techniques—perhaps encouraging more expert group members to create learning models for the less expert.

An important element of minimalist design is to amplify learners' inquiry and sense-making. In a self-paced learning context, one way to do this is to initiate goal-oriented activities where procedural details are sketchy or absent (e.g., Carroll & Carrithers, 1984). At a group level, this might be accomplished by engaging members in distributed activities where no individual holds a complete understanding of how other participants are contributing (Hutchins, 1995). Technical support for inquiry in such a context could be a history system that not only logs members' activity streams but also their rationale. Alternatively one might recruit social mechanisms for provoking group inquiry and reflection, for instance organizing cross-generation or cross-cultural groups that must test and establish common ground in order to collaborate (Rosson & Carroll, 2003; Veinott et al., 1999).

Community knowledge is diverse, distributed, and evolving. The group can access, engage with, and benefit from their shared knowledge to the extent that it is available for application in the right place and at the right time. This implies a requirement for knowledge management—enabling the community to capture, organize and recruit members' knowledge, both tacit and explicit (Davenport & Prusak, 1997). Technical approaches to this might involve member profiles and history, perhaps including an explicit expertise database (Ackerman & Malone, 1990). An alternative is to collect a variety of information about the community's activities (e.g., online objects or files accessed, changes instrumented, messages posted) and offer visualizations or other tools to extract patterns from the data (Hill & Begole, 2003). A social technique might be to initiate and reinforce "pockets of expertise" in the community (Rosson, 2004).

Finally, the social structure of a community leads to an expanded view of problem recognition and recovery. As groups pursue shared goals, the members who are active will vary in the expertise, time, and other resources that they can contribute; the quality or accuracy of their contributions may vary accordingly. The community must be able to monitor, detect, and correct problems experienced by individuals or subgroups, with the result that the group and its activities evolve together. A designer can encourage social support of this sort by enabling flexible ownership and digital editing rights (e.g., the Wiki model; Cunningham & Leuf, 2001), or by providing for reputation or other expertise mechanisms that highlight sources of expertise (and accompanying responsibility; Kelly, Sung & Farnham, 2002).

Minimalist Design for Community Computing

Over the past decade we have studied a variety of issues and Internet technologies in community computing (Carroll, 2002; Carroll & Rosson, 2001; Carroll & Rosson, 1996). We have worked in several contexts, including K-12 education (Isenhour et al., 2000), senior citizens (Carroll et al., 1999) and inter-generational computing (Rosson & Carroll, 2003), civic groups (Carroll et al., in press; Kavanaugh et al., 2003) and non-profit organizations (Merkel et al., 2004). Although our community computing projects were not developed with minimalism as an explicit design goal, together they illustrate features that begin to define a minimalist design space for such efforts.

Figure 2 offers an overview of four projects that we highlight in the current discussion. BLACKSBURG NOSTALGIA was an early Internet application built in collaboration with the town's senior citizens (Carroll et al., 1999). It used interactive forms to collect stories of what it was like to live in this town in the 1950's and 1960's; these stories were often prompted by photographs taken during that era, and were elaborated, refined, and sometimes even corrected through other residents' comments and annotations.

TEACHER BRIDGE is an ongoing participatory design project in which public school teachers build online artifacts and activities in a collaborative space that allows them to share, reuse and build their teaching practices together (Kim et al., 2003). COMMUNITYSIMS was an exploration of cross-generational learning and collaboration in which middle school students worked with senior citizens to raise and discuss community issues using visual simulations (Rosson et al., 2002; Rosson & Carroll, 2003). CIVIC NEXUS is a relatively new project in which we are working with non-profit community groups to help them understand their needs and goals for information technology, as well as how to acquire and sustain the skills for meeting these needs (Farooq et al., 2005; Merkel et al., 2004).

Although each of these projects was motivated by different community computing research goals, they share the fundamental belief that the learning and

use of community computing skills should occur in an informal manner that is contextualized by the needs and resources of the group. In this section we reflect on characteristics of the projects that illustrate how the five minimalist design principles might be applied within a community computing context.
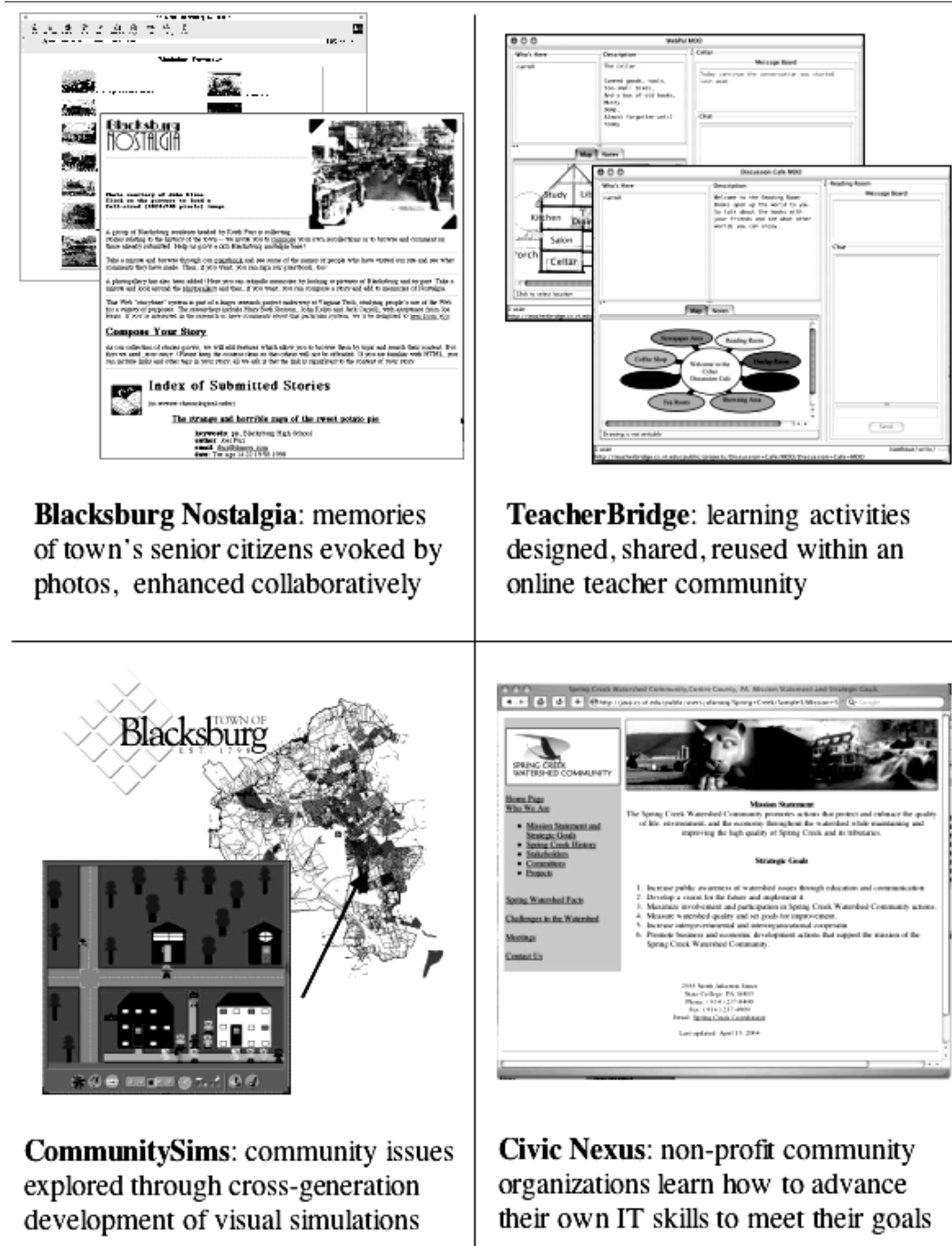

Blacksburg Nostalgia: memories of town's senior citizens evoked by photos, enhanced collaboratively

TeacherBridge: learning activities designed, shared, reused within an online teacher community

CommunitySims: community issues explored through cross-generation development of visual simulations

Civic Nexus: non-profit community organizations learn how to advance their own IT skills to meet their goals

Figure 2. Overview of four community computing projects used as examples.

Embed Learning in Meaningful Activities

In two of our projects, informal learning opportunities developed out of existing activities. This has the advantage of fitting the new computing services into an established set of goal structures and methods. For example, NOSTALGIA was conceived as a web-based enhancement to a story-collection process that was already underway using email and listservs. The meaning of this activity was tied to residents' shared history of interactions and events in a familiar set of places that had evolved over time. TEACHER BRIDGE offers teachers a suite of tools (web editors, shared data tables, electronic whiteboards, etc.) within a collaborative authoring environment. But rather than enrolling teachers in a "collaborative authoring" activity from the start, we tapped their strong existing motivation to create teaching materials. Our strategy was to engage teachers by supporting well-established pedagogical goals, while at the same time enabling a new framework for sharing and collaboration among teachers (Carroll et al., 2003). In this sense, the informal learning about one another's efforts emerged as a by-product of the familiar activity of classroom lesson development.

In contrast, the COMMUNITYSIMS and the CIVIC NEXUS projects illustrate the *discovery* of meaningful activities as an integral part of informal learning. In COMMUNITYSIMS our vision was of cross-generation construction and discussion of simulations that address community issues (e.g., noisy neighborhoods, smoking at school). But we predicted from the start that middle school students and elderly citizens would have differing reactions to this goal; one of our research tasks was to investigate these reactions and find a way to synthesize them into a shared effort. We discovered that the elders understood the project as community education and the students as game-playing; this led us to reconceive the project as a role-playing activity in which elders designed community simulations that were implemented by the students. The synthesized activity was multi-faceted, enabling the rather different participants to at once respect their complementary strengths and to find a comfortable framework for their interactions. The students learned about community issues and the elders learned about the design and implementation of interactive graphical simulations.

In CIVIC NEXUS we have been helping community groups reflect on and understand their own goals with respect to computing (Merkel et al., 2004). For instance in one case a watershed conservancy group had initiated a website development activity by hiring an outside expert. They found that this activity was more complex than they had imagined—the expert built a perfectly lovely website but one that did not convey the group's image of itself. With our help the group has reconceptualized the activity as conveying their mission to the public and is now working on their own version of a website. We showed the group how

to use scenarios to better understand their computing goals, for example taking an abstract concept like "mission" and exploring it through a concrete usage episode.

Allow Learners to Get Started Quickly

Our community computing projects exemplify two complementary techniques for helping groups to make progress quickly. One of these is based on the recognition that group members will vary in their starting point and so will need differing options for participation. The second recognizes that because some members of the group are better prepared for the new technology, the activity should enable them to take the lead so that the group as a whole makes rapid progress.

In our TEACHERBRIDGE project, we recognized that teachers vary considerably in their technology background, but we wanted any motivated teacher to be able to quickly get up to speed in the collaborative environment. Our approach was to provide simple web-based editing tools (building from the wiki paradigm) that operate in parallel with fully interactive Java-based tools for creating and editing web pages, discussions, calendars and so on. Teachers who are comfortable authoring web pages can use the simpler tools to work *on the same objects* as those created by other teachers using the more sophisticated tool suite.

From a group activity perspective, an early measure of progress is the speed with which pre-existing activities can be reconstituted in a new community computing system. For instance, most of the non-profit groups in the CIVIC NEXUS project already have some web presence and they do not want to start over using new tools. Thus we have emphasized the process of reusing existing content (e.g., importing a series of HTML pages into a content management tool). In this way a group can quickly have a rough prototype (one that is also familiar) and focus on editing and enhancement rather than building new content from scratch.

The NOSTALGIA project is simple in its technology but illustrates an important social technique for making progress quickly. We identified senior citizens who had significant Internet expertise and recruited them as our "lead users", ensuring that a critical mass of town memories were contributed and discussed in the first weeks of system use. At the same time, we serves as intermediaries for elders who wanted to contribute but were not ready to use the web interactive system—if they sent a story or annotation to our research team, we would post it on their behalf. We are using a similar strategy in CIVIC NEXUS: in one activity involving courseware construction by high school students, we quickly identified those who had network expertise and oriented them toward installation of server software, while others just as quickly began developing courseware content (web pages).

The COMMUNITYSIMS project also used a form of social scaffolding to move the activity along quickly. Because participants were assigned a role (elders were simulation designers, students were programmers), they had less need to negotiate common ground with their young or elderly partner; the complementary

responsibilities made it clear who should do what. Stereotypes are often seen as limiting and potentially damaging to interpersonal behavior, but in this situation they seemed to streamline the cross-generation interaction, providing a sort of "bootstrapping" for collaborative exchange.

From an informal learning perspective, the COMMUNITYSIMS project faced the risk that the visual simulation programming process would slow down the activity, frustrating participants and sapping their intrinsic motivation. The simulations were built using STAGECAST CREATOR, a visual language that uses visual before-after rules and programming by demonstration to build simulations that play out as a series of connected animations (Smith & Cypher, 1999). We addressed this issue by providing a number of canonical simulations that could be played and quickly enhanced, enabling early progress and reward. During the more extended simulation and construction process, we encouraged design teams to employ a very lightweight "tool" for quick results—paper and pencil for sketching characters and action. One way to see the sketching process is as a low-fidelity prototype that can be developed in a participatory fashion regardless of the partners' technology expertise (Erickson, 1995; Muller, 199?).

Rely on Learners to Think and to Improvise

A fundamental goal of minimalist design is to provoke the learner into an active learning process of inference and problem-solving rather than one of rote learning, procedure following, and repetition. In group activities, this contrast is often discussed by comparing collective behavior that is situated and emergent versus behavior that is pre-defined and regulated (cf. Suchman's [1987] situated action versus Winograd's [1987] action workflow approach).

Our community computing projects are all open-ended in nature. The most prescribed is NOSTALGIA, where we were clearly requesting a certain type of entry (town memories). However even in this case, the tool did not enforce any content or format restrictions, and in fact we observed that contributors learned to use the tool for unexpected purposes (e.g., online greetings). In the COMMUNITYSIMS workshops we provided examples of the sorts of simulations we thought were suitable, but the ideas generated and implemented by the participants were more broad-ranging and creative than our starting set. The younger students were quite content to build characters and actions that had some aspect of fantasy, while the older participants proposed and encouraged "real world" connections to actual places and events taking place in the town. The resulting simulations illustrated an interesting combination of fantasy intermixed with familiar names, buildings, and other details (Rosson & Carroll, 2003).

Some of our community computing tools have features that were explicitly designed to provoke inference and engagement. For instance TEACHERBRIDGE includes "awareness" mechanisms that convey the current status of group efforts.

One of these is a timeline that visualizes whether and how often a project component has received attention, including deadlines that might be looming. Group members can make inferences about activity needs and decide whether and how they should take steps to meet these needs. Similarly, teachers can find, view and test work products created by other teachers, but if a teacher wishes to create a similar artifact she must discover how it was constructed—the system facilitates this by storing information about the object's creator.

In the CIVIC NEXUS project we are exploring a minimalist form of participatory design in which we refrain from co-designing technology solutions. Instead we present multiple possibilities and highlight the tradeoffs associated with each. The goal is to provide enough assistance that the group can actively discuss and choose among options, learning more about their needs, values, and resources in the process (Merkel et al., 2004). At a conceptual level this is analogous to the early work on minimal manuals, where the instructional designer uses open-ended instructions or queries to make learners think about how to solve a problem rather than tell them directly what to do (Carroll & Carrithers, 1984). The method can be conveyed by the metaphor of a *bard*: an actor who stands at the periphery but who tells stories that enhance the group's self-understanding (Carroll & Rosson, in press).

Evoke and Leverage Learners' Prior Knowledge

The knowledge of a group is distributed and evolving, so one objective of minimalist design is to make the group's collective knowledge more apparent and accessible. A good example can be seen in TEACHERBRIDGE: the system records everything that happens, every object that is created or browsed, every change that is made, emails or chat messages sent. For instance, suppose that an interactive discussion room is created by one teacher for her class, reused by a second teacher for a related discussion, and used by a third as a model for a class debate. The knowledge about how to create these educational activities and the consequences of using them for different purposes is distributed across the three teachers. But because the system records all of these interactions and transformations, an interested teacher can access this knowledge. For example she might view a social network of an object's authors and users, and may choose to contact them directly or to explore other class activities they have created.

The stories that residents contributed to NOSTALGIA were a natural mechanism for evoking, elaborating and refining the local history knowledge distributed among community residents (see also Palaver tree paper). We further reminded the senior citizens of their shared knowledge by acquiring, digitizing, and presenting historical photos of buildings and other familiar sites in the town. This is analogous to the evocative role of *sacred places* in community discussions (Erickson, 2000).

As with other minimalist design principles, a designer may use social techniques to engage a group's prior knowledge. In the COMMUNITYSIMS project we knew that the knowledge bases of our cross-generation community members would be diverse—middle school students are familiar with the social and material concerns of 12-14 year olds, whereas elderly residents are familiar with civic concerns, policy tradeoffs, and so on. To ensure that we offered content that would connect with both groups, we carried out a preliminary workshop in which we worked with middle school teachers and elderly citizens to brainstorm simulation topics that would be age-appropriate. For example, the teachers directed us to student-relevant topics like flirting and sexual harrassment, while the older users proposed topics like voter education and neighborhood watch. Because this project was aimed at integrating two segments of the community with rather different knowledge bases, we were careful to provide initial examples that would make contact with each subgroup.

Prevent, Mitigate, and Leverage Errors

Minimalist design research has provided many examples of scaffolding intended to prevent or minimize the damaging consequences of errors—for example *training wheels* that inactivate system function choices known to cause confusion and frustration (Carroll & Carrithers, 1984). Such a technique enables the learner to make a mistake, and to discover that it is an inappropriate step to take, but without suffering any other consequence. In the context of community computing an analogous approach leverages the group's social structure and shared goals to create a social form of scaffolding; other members of the community would be able to help a member having difficulty, even to the extent of correcting an error directly.

The TEACHERBRIDGE environment was designed with this sort of social scaffolding in mind. When a teacher creates a new object, she can decide what other members of her community should be able to view it and who if at all should be able to edit it. If she makes it editable by others, not only can they explore her creations, they can enhance them with their own edits, whether minor fixes like formatting or spelling problems, or more significant enhancements like new embedded objects. Because the system also has extensive support for object histories and versions, these friendly amendments are made with low risk; the original author can always choose to reinstall an earlier version. These mechanisms are descendents of Ward Cunningham's wiki concept (Cunningham & Leuf, 2001) and are designed to create a community culture of shared responsibility, a sort of good citizenship directed at ensuring high quality.

A related example can be seen in the simulation-specific commenting facility in COMMUNITYSIMS. When a simulation is posted for sharing, it is allotted its own mini-discussion thread, a place for other users (or even the author) to post

reactions and suggestions. We discovered that many of the comments posted were aimed at perceived problems or complaints about the simulations, for example suggesting places where the action needs to be smoothed out, or where the message does not come across. In this case it is up to the author to address the problem, but other members of the community assist with problem recognition and possible solutions.

In the CIVIC NEXUS project, problem recognition and recovery is subsumed under the more general concern of technology capacity management. A high level objective in all of our non-profit partnerships is *sustainability*: when the research project is over, the group must be able to maintain (and even further develop) the technology it needs to achieve its goals. Sustainability includes the recognition and recovery from problematic situations (e.g., the server software becomes outdated) but goes beyond this to include management of technology skills and plans. In CIVIC NEXUS we are designing for sustainability by understanding our research objective to be initiation of an ongoing learning *process* rather than development of a *product* or solution.

Discussion

We have used the minimalist design framework to reflect on several of our projects in community computing, illustrating how these projects contain a number of features designed to promote informal learning. We posited that the context of community computing engenders a mix of opportunities and challenges for informal learning; for instance the diversity of the groups and their members' motivation creates opportunities for cross-fertilization and mutual learning, but at the same time makes the learning process relatively unpredictable and distributed across time, people, and locations. The general lack of resources leads to an important requirement for flexible, just-in-time learning mechanisms. The minimalist techniques discussed in this paper and summarized in Table 2 are offered as guidance to designers seeking to support informal learning in a community computing context.

For example, we noted that informal learning must be motivated by and grounded in meaningful behavior. Within a community computing context, we argued that designers' focus should be on *activities* rather than on system functions, and that these activities should have a shared meaning in the community that is multi-faceted and emergent. Designers can promote a shared understanding among community members by helping them to recognize contrasting or complementary views and to synthesize them into a reconceptualization of the group's activity. Designers might facilitate a synthetic process such as this by providing group members with lightweight tools for externalizing and combining one another's differing views.

Design principle	Example minimalist strategies
Meaningful activities	*Social*: Discussion and synthesis of contrasting or complementary goals
	Technical: Lightweight tools (e.g. sketching) for developing a shared understanding
Get started quickly	*Social*: Stereotypical role definitions
	Technical: Multiple levels of commitment; importing content from existing activities
Think and improvise	*Social*: Discussion of tradeoffs and design rationale; teams with diverse membership
	Technical: Pervasive activity awareness
Prior knowledge	*Social*: Content customized to different subgroups and knowledge bases
	Technical: Data about members' creations, usage, and transformations; tools for mining these data
Error recovery	*Social*: Designing for technology capacity management
	Technical: Collaborative editing with versioning; object-specific commentary or annotations

Table 2. Social and technical approaches to minimalist design

The diversity of community groups might seem like a deterrent to rapid progress in a learning situation, but we showed how we were able to leverage the diversity in a team by setting up a role-playing situation in which different members had clear but complementary responsibilities. Similarly we argued that encountering different or even competing viewpoints and capacities in a group can be the inspiration for thinking and improvisation.

We suggested that if designers provide pervasive support for access rights and versioning within a community, they may encourage a culture of shared responsibility in which members help to monitor and correct one another's work. We also argued for a broader view of problem detection and correction, where it is subsumed by the more general concern of technology capacity management.

Note that the strategies summarized in Table 3 have been classified as primarily social or technical in nature. These labels are not intended to be mutually exclusive—a tool used to mine group members' online interaction data is a technical innovation but it clearly is working with information that is social in character. However the contrasting labels do help to express the complementary contributions of technical and social design elements in community computing. This is an important general lesson to draw from our discussion: community

computing should be analyzed as a socio-technical system, and the minimalist design strategies should incorporate both social and technical approaches.

Another general observation about the community computing projects we have reviewed is that all have relied extensively on participatory design methods (Carroll & Rosson, in press). Meaning and intrinsic motivation are an essential ingredient of informal learning and designers will not be able to appreciate community member's concepts and concerns unless they work with them, probe their views, present options, and listen to reactions.

Indeed, our current work in the CIVIC NEXUS project is leading us to a new view of participatory techniques that are appropriate for community computing. The lack of stable governing structures and resources makes the sustainability of any community computing intervention a first order concern (Merkel et al., 2004). If a research project enhances the technological support for a group's activity, what happens when the project is over? Will the group be able to maintain its new technology and activities? Can the group learn and evolve as the technology evolves? If the goal of the participatory engagement is seen as the development of a community computing "system" (or even an activity), chances are that the intervention will not be sustainable in the longer term, for instance as the non-profit group's membership evolves, and its resources come and go. Instead we are coming to believe that a more appropriate design goal is to initiate a *technology learning practice*. A socio-cultural change of this sort may not lead to specific designed artifacts, but it may be resilient to changes in membership, the departure of the research group, and so on.

As designers, we find it difficult to resist a solution-oriented exchange with our community partners—we hear their needs; we have great ideas about how to solve their problems; we want to help! But if instead we carefully listen and reflect back to them what we have heard, we may be able to do even better: we may be able to help them help themselves.

Not surprisingly, our emerging view of participatory design is minimalist itself. We try to identify modest problems that a group can tackle immediately, so that they can see themselves making progress. We encourage them to build from their own knowledge by telling them stories about what we have heard them say, so that they can better understand what they do or do not already know. We avoid giving them detailed instructions for new activities, instead presenting options and tradeoffs for them to consider. When we identify sources of energy (even when they are in the form of a conflict) we reflect these back into the process to inspire thinking and improvisation. Finally we try to guide them toward more effective intra-group communication (e.g., using scenarios) so that they can monitor their own understanding and progress toward goals.

Community computing presents a challenging and rewarding context for design. If our goal is to integrate community members' informal learning about technology into their real world activities, then the sustainability of the learning

process must become a high-priority requirement in our work. Ultimately this may cause us to shift our attention from the design of community computing *systems* to the evolution of community computing *practices.*

Acknowledgments

References

Ackerman, M.S. & Malone, T.W. 1990. Answer Garden: A tool for growing organizational memory. In *Proceedings of the Conference on Office Systems* (pp. 31-39). New York: ACM.

Beamish, A. 1995. *Communities On-Line: Community-Based Computer Networks.* Masters Thesis, Department of Urban Studies and Planning, MIT.

Bertelsen, O.W. & Bødker, S. 2003. Activity Theory. In J.M. Carroll (Ed.), *HCI Models, Theories, and Frameworks: Toward a Multidisciplinary Science* (pp. 291-324). San Francisco: Morgan Kaufmann.

Carroll, J.M. 1990. *The Nurnberg Funnel: Designing Minimalist Instruction for Practical Computer Skill.* Cambridge, MA: MIT Press.

Carroll, J.M. (Editor) 1998. *Minimalism beyond "The Nurnberg Funnel".* Cambridge, MA: M.I.T. Press.

Carroll, J.M. & Carrithers, C. 1984. Blocking learner errors in a training wheels system. *Human Factors, 26(4),* 377-389.

Carroll, J.M. & Kay, D.S. 1985. Prompting, feedback, and error correction in the design of a scenario machine. In B. Curtis & L. Borman (Eds.), *Proceedings of Human Factors in Computing Systems: CHI '85* (pp. 149-154). New York: ACM.

Carroll, J.M., Choo, C.W., Dunlap, D.R., Isenhour, P.L., Kerr, S.T., MacLean, A., & Rosson, M.B. 2003. Knowledge management support for teachers. *Educational Technology Research and Development*, 51(4), 42-64.

Carroll, J. M., & Rosson, M. B. 1987. The paradox of the active user. In J.M. Carroll (Ed.), *Interfacing Thought: Cognitive Aspects of Human-Computer Interaction*. Cambridge, Mass: MIT Press (pp. 80-111).

Carroll, J. M. & Rosson, M. B. 1991. Deliberated evolution: Stalking the View Matcher in design space. *Human-Computer Interaction, 6*(3&4), 281-318.

Carroll, J.M. and Rosson, M.B. 1996. Developing the Blacksburg Electronic Village. *Communications of the ACM,* 39 (12), 69-74.

Carroll, J.M. & Rosson, M.B. 2000. School's Out: Supporting authentic learning in a community network. *IFIP Conference on Information Technology at Home* (Wolverhampton, United Kingdom, June 28-30).

Carroll, J.M. and Rosson, M.B. 2001. Better home shopping or new-democracy: Evaluating community network outcomes. *Proceedings of CHI 2001: Conference on Human Factors of Computing Systems.* (Seattle, WA; March 31-April 5). NY: ACM, 372-379.

Carroll, J.M. & Rosson, M.B. 2003. A trajectory for community networks. *The Information Society*, 19(5), 381-393.

Carroll, J.M. & Rosson, M.B. (in press). Participatory design of information systems. To appear in *Human-Computer Interaction in Information Systems Design.*

Carroll, J.M., Rosson, M.B., Isenhour, P.L., Ganoe, C.H., Dunlap, D., Fogarty, J., Schafer, W., & Van Metre, C. 2001. Designing our town: MOOsburg. *International Journal of Human-Computer Studies,* 54, 725-751.

Carroll, J.M., Rosson, M.B., VanMetre, C.A., Kengeri, R., & Darshani, M. 1999. Blacksburg Nostalgia: A Community History Archive. In M.A. Sasse & C. Johnson (Eds.), *Proceedings of Seventh IFIP Conference on Human-Computer Interaction INTERACT 99* (Edinburgh, August 30 - September 3). Amsterdam: IOS Press/International Federation for Information Processing (IFIP), pages 637-647.

Cunningham, W., & Leuf, B. 2001. *The Wiki Way: Collaboration and Sharing on the Internet*. Reading, MA: Addison-Wesley.

Davenport, T.H. & Prusak, L. 1997. *Working Knowledge: How Organizations Manage What They Know*. Cambridge, MA: Harvard Business School Press.

Dewey, J. 1933. *How We Think: A Restatement of the Relation of Reflective Thinking to the Educative Process*. Boston: D.C. Heath.

Engeström, Y. 1987. *Learning by Expanding.* Helsinki: Orienta-Konsultit.

Erickson, T. 1995. Notes on design practice: Stories and prototypes as catalysts for communication. In J.M. Carroll (Ed.), *Scenario-Based Design: Envisioning Work and Technology in System Development* (pp. 37-58). New York: John Wiley & Sons.

Erickson, T. 2000. Lingua Francas for design: Sacred places and pattern languages. *Proceedings of DIS 2000* (pp. 357-368). New York: ACM.

Farooq, U., Merkel, C., Nash, H., Rosson, M.B., Carroll, J.M. & Xiao, M. 2005. Participatory design as apprenticeship: Sustainable watershed management as a community computing application. *Hawaii International Conference On System Sciences: HICSS 38.* Kona, HI, January 2005.

Granovetter, M. 1973. The strength of weak ties. *American Journal of Sociology, 78*:1360-80.

Isenhour, P., Carroll, J.M., Neale, D., Rosson, M.B., & Dunlap, D. 2000. The Virtual School: An integrated collaborative environment for the classroom. *Educational Technology & Society*, 3(3), 74-86.

Hill, R. & Begole, J. Activity rhythm detection and modeling. In *CHI 2003 Extended Abstracts.* New York: ACM.

Hutchins, E. 1995. *Cognition in the Wild,* Bradford: MIT Press.

Kavanaugh, A. Reese, D.D., Carroll, J.M., & Rosson, M.B. 2003. Weak ties in networked communities. In M. Huysman, E. Wenger and V. Wulf (Eds.) *Communities and Technologies 2003* (pp. 265-286). Amsterdam: Kluwer Academic Publishers.

Kavanaugh, A., Carroll, J.M., Rosson, M. B., Reese, D. D. & Zin, T.T. 2005. Participating in civil society: The case of networked communities. *Interacting with Computers* (in press).

Kelly, S.U., Sung, C., & Farnham, S. 2002. Designing for improved social responsibility, user participation and content in online communities. In *Proceedings of CHI 2002* (pp. 391-398). New York: ACM.

Kim, K., Isenhour, P.L., Carroll, J.M., Rosson, M.B., & Dunlap, D.R. 2003. Teacher Bridge: Knowledge management in community networks. *Home and Office Information Technology: HOIT3.* April 2003, Irvine, California.

Kuutti, K. & Arvonen, T. 1992. Identifying potential CSCW applications by means of Activity Theory concepts: A case example. *Proceedings of CSCW 1992* (pp. 233-240). New York: ACM.

McGiveney, V. 1999. *Informal Learning in the Community. A Trigger for Change and Development*. Leicester: NIACE.

Meij, H. van der & Carroll, J.M. 1995. Principles and heuristics for designing minimalist instruction. *Technical Communication, 42,* 243-261.

Merkel, C.B., Clitherow, M., Carroll, J., & Rosson, M. 2004. *Sustaining computer use and learning in community computing contexts: Making technology part of "Who They are and What They Do".* Paper presented at CIRN 2004: Sustainability and Community Technology, 9/29-10/01, 2004. Monash University, Prato, Tuscany, Italy.

Merkel, C. B., Xiao, L., Farooq, U., Ganoe, C. H., Lee, R., Carroll, J. M., & Rosson, M.B. 2004. Participatory design in community computing contexts: Tales from the field . *Proceedings of PDC 2004* (pp.1-10). New York: ACM Press.

Muller, M.J., Wildman, D.M., & White, E.A. 1993. 'Equal opportunity' PD using PICTIVE. *Communications of the ACM*, 36(4), 54-66.

Putnam, R. 2000. *Bowling Alone: The Collapse and Revival of American Community*. New York: Simon and Schuster.

Rosson, M.B. 2004. Computer-supported cooperative work: Vignette. In *Berkshire Encyclopedia of Human-Computer Interaction.*

Rosson, M. B. & Carroll, J. M. 1996. Scaffolded examples for learning object-oriented design. *Communications of the ACM*, 39(4), 46-47.

Rosson, M.B. and Carroll, J.M. 2003. Learning and collaboration across generations in a community. In M. Huysman, E. Wenger & V. Wulf (eds.), *Communities and Technologies 2003* (pp. 205-225), The Netherlands: Kluwer Academic Publishers. (Amsterdam, September 2003).

Rosson, M. B., Carroll, J. M., & Bellamy, R. K. E. 1990. Smalltalk scaffolding: A case study in Minimalist instruction. In J. C. Chew & J. Whiteside (Eds.,) *Proceedings of Human Factors in Computing Systems, CHI'90 Conference* (pp. 423-430). New York: ACM.

Rosson, M.B., Carroll, J.M., Seals, C., & Lewis, T. 2002. Community design of community simulations. *Proceedings of Designing Interactive Systems: DIS 2002* (pp. 74-83). New York: ACM.

Rosson, M. B., Carroll, J. M., & Sweeney, C. 1991. A View Matcher for reusing Smalltalk classes. In S. P. Robertson, G. M. Olson, & J. S. Olson (Eds.), *Proceedings of Human Factors in Computing Systems, CHI'91 Conference* (pp. 277-284). New York: ACM.

Rosson, M.B. & Seals, C. 2001. Teachers as simulation programmers: Minimalist learning and reuse. In *Proceedings of CHI 2001* (31 March - 4 April, Seattle, WA), pp. 237-244.

Smith, D.C. & Cypher. A. 1999. Making programming easier for children. In Druin A., ed. *The Design of Children's Technology,* Morgan Kaufmann, San Francisco, 1999, pp. 201-222.

Suchman, L. 1987. *Plans and Situated Action: The Problem of Human-Machine Communication.* Cambridge, UK: Cambridge University Press.

Uncapher, W. 1999. New communities/new communication: Big Sky Telegraph and its community. In M. Smith & P. Kollock, eds., *Communities in Cyberspace.* Routledge.

Veinott, E.S., Olson, J., Olson, G.M., & Fu, X. 1999. Video helps remote work: Speakers who need to negotiate common ground benefit from seeing each other. *Proceedings of CHI 1999* (pp. 302-209). New York: ACM.

Vygotsky, L.S. 1962. *Thought and Language.* Cambridge, MA: MIT Press.

Winograd, R. 1987. A language/action perspective on the design of cooperative work. *Human-Computer Interaction*, 3(1), 3-30.