Answer Garden 2: Merging Organizational Memory with Collaborative Help

Mark S. Ackerman
David W. McDonald
Department of Information and Computer Science
University of California, Irvine
Irvine, CA 92717
{ackerman, dmcdonal}@ics.uci.edu
http://www.ics.uci.edu/CORPS/ackerman.html

ABSTRACT
This research examines a collaborative solution to a common problem, that of providing help to distributed users. The Answer Garden 2 system provides a second-generation architecture for organizational and community memory applications. After describing the need for Answer Garden 2's functionality, we describe the architecture of the system and two underlying systems, the Cafe ConstructionKit and Collaborative Refinery. We also present detailed descriptions of the collaborative help and collaborative refining facilities in the Answer Garden 2 system.

KEYWORDS: computer-supported cooperative work, organizational memory, community memory, corporate memory, group memory, information refining, information retrieval, information access, information systems, CMC, computer-mediated communications, help, collaborative help, CSCW

INTRODUCTION

Many user communities have a problem with delivering help and general assistance. Unfortunately, the user is often left to sift through reams of documentation, find his way through mail archives, or pursue answers through trial and error. Normally, one attempts to examine the documentation or other help sources, and then wanders out into a hallway in search of friendly colleagues.

The problem becomes acute, however, in distributed communities. We take for our example the astrophysics community, although this problem exists in most scientific communities. In the astrophysics community, the users may be spread across the world, they may work in isolation, and they may have need of relatively specialized help. What we would like is a surrogate for this hallway talk. Such a solution must avoid the broadcast problem of flooding everyone's electronic mail basket with thousands of questions. Instead, this work reports on a system to narrow-cast a question to the appropriate others, whether those others are experts or colleagues.

Computer Supported Cooperative Work '96 , Cambridge MA USA
© 1996 ACM 0-89791-765-0/96/11 ..$3.50

This research, then, examines a collaborative solution to a common problem. Earlier work, a system called Answer Garden, allowed organizations to develop databases of commonly asked questions that grow "organically" as new questions arise and are answered. The subsequent Answer Garden 2, the focus of this paper, continues this work. It is a second-generation architecture for the same design problem, investigating some of the issues encountered in field studies of the original system. The new architecture provides a customizable and adaptable set of software components that allow a variety of organizational and informational configurations. Furthermore, it offers a generalized solution to the problem of finding help for any information system. We report here on the new architecture and its responses to the context and authoring issues.

The paper begins with a brief introduction to the help and memory problems, as well as a brief overview of the original Answer Garden application and its field study results. Answer Garden 2 is then introduced. After an explanation of its architecture, the paper analyzes two particular features of Answer Garden 2. These two features, collaborative help and collaborative refining, are explained at length. Collaborative help mechanisms provide the necessary context for information, and collaborative refining mechanisms provide support for authoring. The paper concludes with a survey of related CSCW systems and some conclusions about these design considerations.

FRED'S PROBLEM

Fred (not his real name) is an astrophysicist at the Harvard-Smithsonian Astrophysical Observatory. He, like many scientists, does not want to know anything about software systems or his hardware. He wants to do his scientific work, free of the multitude of computer problems that seem to get in the way.

In the "old days," everyone sat around in a common room, using their computer consoles with the mini-computer. If Fred had a question, he could ask one of the half-dozen to dozen colleagues and programmers sitting in the room.

Everyone had to hear the answer, so the community learnt from the problems of each individual.

Now, Fred sits in his office with his workstation near his desk. It is quieter, but much more isolated. If he has a problem or a question, he can look through the documentation or send electronic mail for help. If he sends electronic mail, he may not get an answer from the programmers for some unknown period of time, or he may be given a response that makes him feel stupid for not knowing the answer. Often he resorts to wandering through the hallways, looking for people who might know the answer. He then tries various possibilities until he finds a solution or he gives up.

Any community, institution, or organization of any size often has a problem with answering questions in a timely manner. Yet, solving problems and completing tasks are often dependent on obtaining timely answers to specific questions.

Fred's problem is the dual problem of help and of collective memory. We will use the term *collective memory* to denote the common attributes of organizational, institutional, and community memory. (The term has a related, but slightly different meaning in the historiographical and critical literatures, but there is no better term to denote memory in a range of collectivities.)

Within an organization or community, individuals' information seeking requires finding the right part of the collective memory. Typically, collective memories include information repositories (e.g., information databases, filing cabinets, documents). It can also include people (e.g., other organizational personnel) [25]. The collective memory to which Fred has access includes at least the documentation, the system programmers, and his colleagues. However, he may have great trouble finding the right piece of the collective memory that has the answer he needs. In other words, his access to the collective memory should be augmented.

Answer Garden and Fred's problem

Previous work, reported in [4] and [2], considered one way of doing this augmentation. This work revolved around a system called Answer Garden. Field studies of its use uncovered a number of important problems in providing collective memory and help to users such as Fred. Before discussing these problems, and our subsequent investigations, it will be useful to briefly describe Answer Garden. This application still plays an important part in our current work.

Answer Garden supports organizational memory in two ways: by making recorded knowledge retrievable and by making individuals with knowledge accessible. In the standard configuration of Answer Garden, users seek answers to commonly asked questions through a set of diagnostic questions or other information retrieval mechanisms. Figures 1 and 2 show Answer Garden reimplemented in the World Wide Web. (Other, third-party versions exist in the Web [22] and in Lotus Notes.) Diagnostic questions guide the user through Web pages. Alternatively, the user may use a number of other information retrieval mechanisms to find the pages that may contain the answer.

If the user cannot find an answer or the answer is incomplete, the user may ask the question through the system. (This is the result of the user pressing the "I'm Unhappy" link in Figure 1.) In the original Answer Garden, the system would then route the question to an appropriate human expert. (This has been changed in Answer Garden 2 as will be discussed below.)

In the original Answer Garden, the expert would then answer the user through electronic mail. If the question was a common one, the expert could insert the question and its answer back into the information database. Thus, users were not limited to the information in the system; if the information was not present, they could tap the organization's experts. As a result, the organization would gain a corpus of information, an organizational memory. Users could obtain expert advice without a high organizational cost. Other interesting properties of the system are discussed in [2].

Open research issues

Field studies of Answer Garden's use ([2], [3]) uncovered a number of issues. While the system was held to have worked, two issues were uncovered that are critical to the success of similar memory or help systems:

- Tying the social network into the system in a more natural manner. Answer Garden's dichotomy between experts and users was problematic. While there was nothing in the underlying technology to force this dichotomy, it was a simplifying assumption in the field study to have separate user and expert groups. Real collectivities do not function this way. Most people range in their expertise among many different skills and fields of knowledge. Fred knows things about systems and his tasks, even though he may not be able to answer specific questions. We would like to allow everyone to contribute as they can, promoting both individual and collective learning.

  However, mechanisms to allow each person to contribute must not overwhelm the other people who use the system. For example, broadcasting each question to every person in an organization or community will fail. AG2 offers several mechanisms to ameliorate the overload problem while allowing and providing for a range of expertise.

- Providing for the contextualization of answers, thus providing for the user's understanding of an answer. In the Answer Garden field study, most users either did not need contextualized information or were able to contextualize it themselves. However, a significant portion of the participants did need more context.

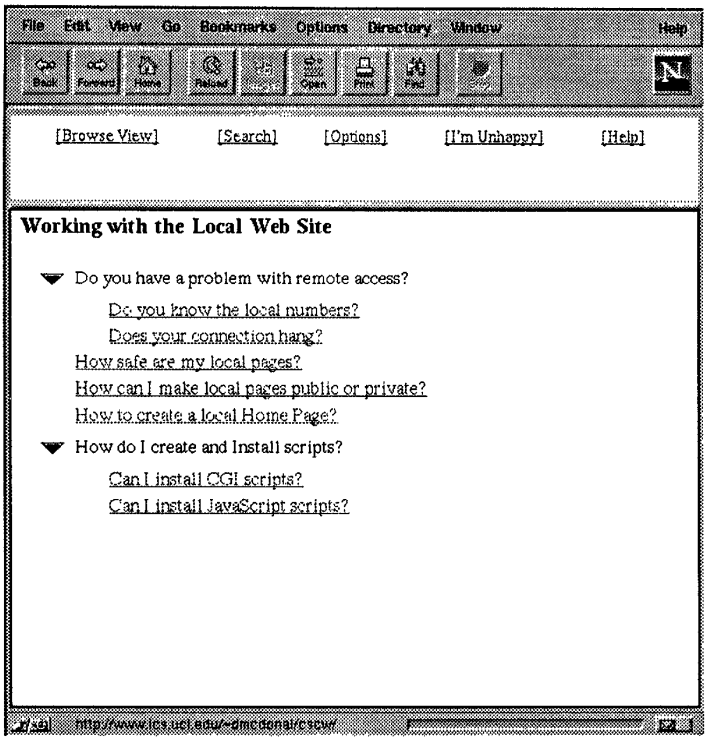


Figure 1: Answer Garden functionality in the Web

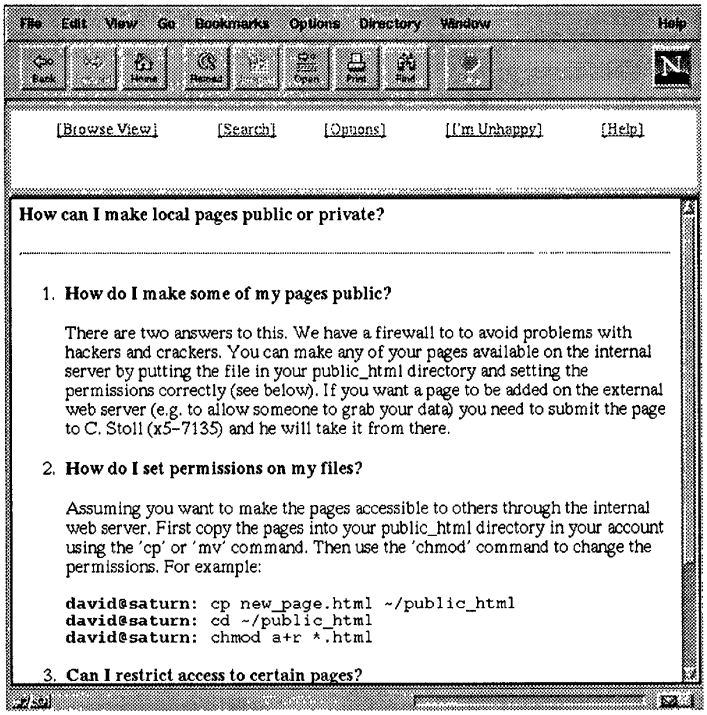


Figure 2: An Answer Garden answer page

In Fred's case, the answer to a question may be present in the documentation. However, he may lack the required expertise to infer an answer or to even use an explicit answer without additional situational information.

Providing the proper context is, unfortunately, difficult. We will return below to one way of potentially providing this context at low cost. Our mechanism also ameliorates the problem of providing answers at the right level and length of explanation.

- Easing the authoring burden. To obtain answers, the cost of authoring must be minimized. Furthermore, authoring answers, as an individual activity, promulgates the distinction between experts and everyone else. The composing content of answers takes as long as any writing takes, but we may be able to ease the mechanics of the process.

One might expect these issues to become increasingly problematic as the information becomes non-technical or the users become less sophisticated in the domain. For example, only astrophysicists can understand the scientific analysis tasks that create their questions about software systems. Astrophysicists will vary in their computer expertise, but few wish to spend time inferring the answer from substantial system documentation before continuing with their analysis tasks. And, the programmers who must currently compose the answers may not even understand the domain or its tasks.

Additionally, the field studies uncovered a number of technical issues, such as the need to use varying "front-end" systems such as the Web or Notes, to consider additional methods of finding experts, and to find better ways of maintaining the information database. These technical issues and the above social issues led us to reconsider the architectural design.

ANSWER GARDEN 2 (AG2)

Answer Garden 2 (AG2) consists of a second generation system architecture for organizational memory and collaborative help support. There are several advantages to this architecture.

First, the design cleanly separates the front-end of Answer Garden (i.e., the user client) from back-end needs. More importantly, it also decomposes the Answer Garden functionality into a set of distributed software services. This provides a high level of organizational flexibility; the services can be mixed and matched in order to provide additional flexibility. For example, by attaching an anonymity service, users of the system can send their questions anonymously. By attaching an anonymity service at another point in the distributed architecture, the experts answering the questions can also be anonymous. Or by not having an anonymity service at all, all users and experts can be known to one another.

Finally, the change in architecture makes much of the help functionality *possible from any information system.* This work, then, is generalizable to any information system.

System components

AG2 is built upon two underlying systems, both of which provide a set of services. These services create the collaborative help and collective memory functionality. The two underlying systems are:

- The Cafe ConstructionKit (CafeCK). CafeCK is a CSCW toolkit for supporting sociality and information use in collaborative environments [6]. CafeCK provides a set of reusable objects that include message transport for asynchronous and synchronous communication (including a Zephyr-like system, NetNews, and email), parsing for a variety of semi-structured protocols, private and public channels for narrowcast communication, message filters, and message retrieval by a variety of semi-structured methods. By selecting from the set of available components (or by extending it) and by writing a simple Tcl program, an application writer can create a set of distributed processes to handle information retrieval, information access, or electronic communications. CafeCK is implemented in C++, Tcl, and Tk.
- Collaborative Refinery (Co-Refinery). Co-Refinery provides mechanisms for handling individual and joint information spaces. Central to Co-Refinery is the ability to individually and collaboratively view and manipulate Answer Gardens and other information

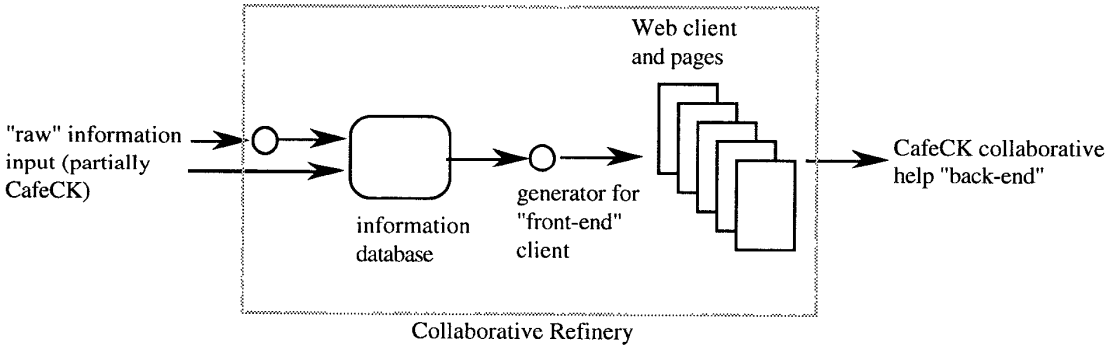
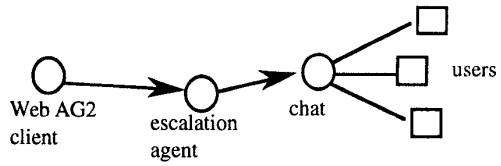
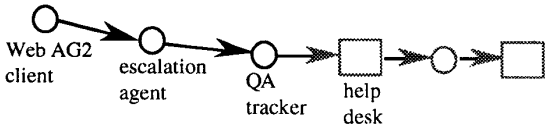


Figure 3: Answer Garden 2 (AG2) architecture



(a) The user's first attempt to get an answer goes to a chat channel.



(b) The user's *j*th attempt to get an answer gets escalated to a help desk.

Figure 4: Two possible escalations for a question

collections. It is especially useful in situations where one wants to refine and distill collections of materials as shared artifacts. It will be described extensively below.

Co-Refinery components include objects for managing a collection archive of materials, constructing and maintaining a database of relationships for those materials, and generating a suitable presentation. Output from Co-Refinery's presentation generator can be HTML, Notes documents, files, or e-mail. Co-Refinery is implemented in C++, and the Web portion relies upon HTML 3.0 and Netscape HTML extensions.

These two components are used together as in Figure 3. Raw information comes into the collection archive through CafeCK processes (such as News filters), by being explicitly sent to the archive through e-mail, or through filtering agents. It may be partially processed, and then is moved into the information database. At snap-shots or upon explicit queries (depending on a site's tailoring of AG2), the materials are built into Web pages, Notes documents, or flat files. In turn, the AG2 Web or Notes clients can send mail to CafeCK back-end processes that then handle the details of obtaining help. These CafeCK help processes will be described next.

COLLABORATIVE HELP

The problem as a duality

AG2's "back end" can be viewed either as a collective memory system or as a collaborative help system. (We use *collaborative help* to denote those help systems that use people as information sources, for example, through Computer-Mediated Communication systems.) Each of these views is the dual of the other. By duals, we invoke the language of linear programming, where two forms exist for each particular problem. Both forms are valid, and users are free to solve the form that provides them with the most analytical tractability. By considering the "back-end" organizational memory problem in terms of its dual, collaborative help, we believe we have found mechanisms for reducing the context problem.

Above, it was noted that an open research issue was how to alleviate the users' need for contextualized information in solving their problems and finishing their tasks. This issue can be ameliorated by using collaborative help in a controlled manner. Collaborative help functionality also provides help to users at their own explanation level and potentially with iterative diagnosis.

Staying local

Providing help from other people -- such as colleagues on the same hall or other group members -- allows people to seek help first from the people most likely to know the local context. Colleagues can judge a person's abilities, expertise, and situation, and can try to provide suitable information to solve the person's problem. Local participants are also more likely to information, since personal social ties are key motivators in providing assistance [7, 19].

Always asking one's colleagues is, however, problematic. First, it is still costly to ask other people. AG2's repository of previously-asked questions and frequently-required information, however, attempts to reduce that problem. More importantly, one's colleagues may not know the answer. While staying local is important, it can also be organizationally dysfunctional [10] when there is no local expert available. In these situations, a means for escalating answers past the local group is required.

Escalation

Using the facilities of CafeCK, we were able to simply construct an escalation agent for questions in AG2. This component allows the user to decide what to do if the question is not answered. It allows the user to consider whether to get answers from chat systems, bulletin boards, software agents, or other people.

The typical way that we envision the system being used is to gracefully escalate the help request until it can be answered. Because the escalation agent is a CafeCK process, the escalation can be quite flexible. The agent is currently programmed to follow organizational rules on the order of escalation, although this is under user control. It would be a simple matter to change this to provide different organizational rules, complete user control, or even heuristics (such as avoiding the chat facility when no other users are logged into their machines). No doubt other mechanisms could be found; this is a potential research question.

In our prototype, the user poses a question through his application. In the example of Fred, the user client is an AG2 front-end, but it can be any application that has asynchronous or synchronous communication capabilities. The application merely connects to a CafeCK process through, for example, e-mail. This CafeCK process, the escalation agent, is semi-autonomous, since it can be triggered either by the user or automatically.

DOCKET ALARM

Explore Litigation Insights

Docket Alarm provides insights to develop a more informed litigation strategy and the peace of mind of knowing you're on top of things.

Real-Time Litigation Alerts

Keep your litigation team up-to-date with **real-time alerts** and advanced team management tools built for the enterprise, all while greatly reducing PACER spend.

Our comprehensive service means we can handle Federal, State, and Administrative courts across the country.

Advanced Docket Research

With over 230 million records, Docket Alarm's cloud-native docket research platform finds what other services can't. Coverage includes Federal, State, plus PTAB, TTAB, ITC and NLRB decisions, all in one place.

Identify arguments that have been successful in the past with full text, pinpoint searching. Link to case law cited within any court document via Fastcase.

Analytics At Your Fingertips

Learn what happened the last time a particular judge, opposing counsel or company faced cases similar to yours.

Advanced out-of-the-box PTAB and TTAB analytics are always at your fingertips.

API

Docket Alarm offers a powerful API (application programming interface) to developers that want to integrate case filings into their apps.

LAW FIRMS
Build custom dashboards for your attorneys and clients with live data direct from the court.

Automate many repetitive legal tasks like conflict checks, document management, and marketing.

FINANCIAL INSTITUTIONS
Litigation and bankruptcy checks for companies and debtors.

E-DISCOVERY AND LEGAL VENDORS
Sync your system to PACER to automate legal marketing.