CAAD: An Automatic Task Support System

Tye Rattenbury and John Canny
Berkeley Institute of Design
Computer Science Division
University of California Berkeley
{rattenbt,jfc}@cs.berkeley.edu

ABSTRACT

Recent HCI research shows strong interest in task management systems (e.g. [19, 27]) that support the multi-tasked nature of information work [13]. These systems either require users to manually create and maintain task representations, or they depend on explicit user cues to guide the creation and maintenance process. To access and use the task representations in these systems, users must also specify their current task. This interaction overhead inhibits the adoption of these systems. In this paper, we present a novel approach to task management that automates the creation and maintenance of task representations. Our system supports the user by making commonly used information more "ready-at-hand" through an intuitive visualization of their task representations. Users can correct and organize their task representations by directly manipulating the visualization; however, this interaction is not required. We describe a feasibility study that demonstrates the actual utility (in terms of overhead reduction) and perceived utility of our system.

Author Keywords

Activity, task, context, information work.

ACM Classification Keywords

H.5.2 User Interfaces, I.2.1 Applied Artificial Intelligence

INTRODUCTION

Recent HCI research has shown strong interest in tools for information workers (people whose primary work function is to create, share, and analyze information). As many studies have shown, information work is characterized by multiple ongoing, often disjoint, tasks [3, 5, 13]. It follows that many problems that arise in the day-to-day lives of information workers relate to task management. Applications in traditional computing environments provide poor support for information workers because they are strikingly unre-
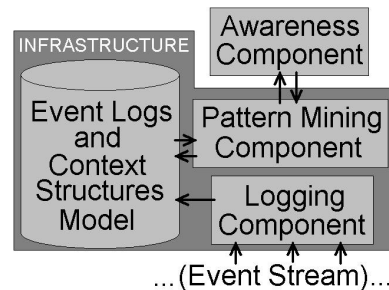

Figure 1. CAAD system architecture. Arrows indicate data flow. Boxes distinguish separate modules.

sponsive to the dynamic nature of task management [10, 20]. Recent attempts to improve support for task management have focused on information management and on organizing representations of ongoing tasks [2, 14, 16, 19, 20, 21, 24, 25, 27, 28]. However, most of these systems suffer from one or both of the following issues: (1) they require too much *overhead* on the part of the user (e.g. expecting users to specify the structure of their tasks), or (2) they lack *contextual awareness* of the user's tasks (e.g. what information is task-relevant – see the Conceptual Background for more detail). In this paper, we present a novel approach to task management that addresses these issues by automating the creation and maintenance of task representations. We also describe a feasibility study that (1) demonstrates the perceived utility of our system in supporting information and knowledge work and (2) verifies the reduced overhead of our system relative to manual task management systems.

Our approach is implemented in a system we call CAAD (Context-Aware Activity Display; rhymes with 'made'). CAAD minimizes user overhead by automatically gathering explicit cues about what the user is doing and then processing these cues to infer the implicit context of the user's activities/tasks. The explicit cues gathered by the logging component of CAAD (Figure 1) are in the form of computer interaction events – e.g. the use of a file, the browsing of a web page, or the execution of an application. Once per day, CAAD applies a custom pattern mining algorithm to logs of these explicit cues. This algorithm detects structures in the user's actual work-flow that encode the implicit context of the user's work activities – i.e. the sets of task-relevant information and people. We will refer to these sets as **context structures**.

1

Most importantly, the awareness component of CAAD (Figure 1) leverages its context-awareness to support the user in two ways. First, it makes real-time (every 30 seconds) predictions on what information items (documents, web pages, etc.) are most relevant to what the user is currently doing. These predictions are an *online* calculation that leverages the *offline* calculation of the user's context structures (see the System Description section for more discussion on this distinction). The higher the predicted relevance of an information item, the more prominently it is displayed (Figure 3) – making it easier to access. Second, the awareness component of CAAD displays information items in groupings that explicitly reveal the set of context structures that it has inferred. By displaying this information, CAAD provides a mechanism for users to become more reflective about the organization of their work behavior. Users can also edit the groupings to better align CAAD's model of their work activities with their own. However, as our study demonstrates, this editing overhead is lower than in a fully-manual task management system.

The rest of this paper is organized as follows. First we describe motivating scenarios that both ground our notion of supporting information workers and highlight the functionality of CAAD. Then we cover related work and discuss some social and psychological theories that support the design of CAAD. With the necessary background covered, we then describe CAAD's architecture. Next, to validate the design of CAAD, we present the results of a field study. Finally, we conclude and propose some future work.

MOTIVATING SCENARIOS

In this section we introduce two motivating scenarios inspired by dialogue from informal interviews with information workers conducted during formative design stages. These scenarios both ground what we mean by "support computer-based information workers" and highlight CAAD's actual functionality.

1. *Information access scenario*. Kate is a knowledge worker writing the related work section of a project report. While writing she realizes that she needs to cite a paper she read at the start of this project, a few months earlier. However, she cannot remember the title or author of the paper, or the exact start date of her project.

2. *Work awareness scenario*. Paul is a project manager coordinating multiple projects teams. To strengthen each project, he has been actively mediating dialogue between members of different teams. To assess which mediation strategies are most effective, Paul has so far relied on informal interviews with members of each project team. However, he is actively searching for more accurate and lighter-weight measures of impact.

In both of these scenarios, the overhead and lack of context issues are key concerns. In the information access scenario, the overhead of having to perform an ill-specified search will likely result in Kate loosing the mental context of writ-

ing. Moreover, the success of this search will depend on Kate's ability to translate the context of her past usage of the document into concrete query terms or constraints. If Kate had been using CAAD since the start of her project, then all of the documents that are contextually relevant to her project report would be prominently displayed (Figure 3). Through the display, Kate could list the relevant documents in the grouping (i.e. context structure) associated with her project; likely recognizing the one she was looking for from its title.

In the work awareness scenario, Paul currently relies on informal interviews for gathering data. These interviews create overhead for Paul and for the project members who must suspend their normal work flow to be interviewed. With the appropriate context data around each project, updated as the projects progress and evolve, Paul could rely on simple similarity measures to detect whether projects are influencing one another or not. For example, if each project member was using CAAD, Paul could measure the number of relevant documents that were shared between projects before he intervened as well as after. If his mediation was effective, the number of shared documents should increase.

These scenarios indicate the type of support we expect CAAD to perform: it should maintain contextual awareness of the user's various activities and use this awareness to (1) minimize the amount of overhead in accessing relevant information and (2) track and reveal the state of the user's various activities.

RELATED WORK

In this section we consider systems that share our overall goal of supporting computer-based information workers. We group these systems into two categories: those that depend entirely on manual input by the user and those that function in a semi-automatic way, requiring only guiding input by the user.

Systems falling into the manual category include Unified Activity Manager [19], Activity Explorer [21], SphereJuggler [20], Activity Based Computing [2], GroupBar [25], Rooms [14], and TaskGallery [24]. Because these systems require direct input from the user, they often only capture the text (as opposed to context) of their user's work practice. To overcome this, many of these systems rely on generic templates to pre-populate task representations (with the usual difficulties of finding representations that are not too generic). CAAD differs from these systems by automatically generating its task representations (as context structures) from logs of low-level, interaction events.

Semi-automatic systems include TaskTracer [27], UMEA [16], and Kimura [28]. The primary input required from users in these system is to indicate, in real-time, what task they are working on. However, people often have trouble labeling and delimiting new tasks and, more importantly, often forget to declare task switches. CAAD overcomes these issues by automatically generating its task representa-

tions and by allowing these representations to evolve as tasks change. We note that CAAD also supports user editing of within task and between task structures. This editing functionality gives users the ability to both correct the context structures detected by CAAD and to organize the context structures into more meaningful arrangements.

CONCEPTUAL BACKGROUND

CAAD's success depends on its ability to accurately detect and track context structures. In this section, we discuss social and psychological theories that argue for the existence of these structures and facilitate the derivation of a general set of context structure characteristics. These generic characteristics are the basis for the data-mining algorithm that CAAD uses to detect and track the context structures of a user's activities.

We draw primarily from Activity Theory (AT) [18, 22, 23]. Activities are the key structure in AT. They are composed of a subject, tools and an objective. The subject is the person, or persons, motivated to carry out and achieve the objective of the activity. The actions performed in an activity are mediated by tools. Tools include everything from found objects like sticks to manufactured objects like hammers to abstract, non-physical objects like words and ideas [18]. In terms of CAAD, users are subjects and documents, folders, applications, and email addresses are tools.

Activities are generally long-term structures whose stability derives from their motivating objective. In working on an activity, however, people tend to focus on shorter-term goals. These goals organize the actions that people perform – e.g. sending an email, cutting a piece of wood, or answering a question. Both actions and the activities they service involve a fairly stable set of subjects (i.e. people) and tools [18]. This stable set of people and tools constitutes the context structure of the user's action and activity. CAAD searches for these stable sets in the event logs it gathers.

Notice we are not claiming that context structures (defined as a stable set of people and tools) reflect a complete definition of context in all its varied uses. Rather, the set of tools and people routinely brought together by subjects acting in the world is one type of context: one that is both computationally feasible to acquire (due to its relative temporal stability) and that adheres to the phenomenological aspects of context [7].

There are many theories related to AT that provide additional perspectives and insight on the nature of context structures. These include Actor Network Theory [17], Distributed Cognition [15], social foci and social network theory [29], genre theory [1, 26] and Goffman's notions of frames [12]. In all of these theories, people, and generally also artifacts and tools, are brought together in the course of everyday action. Although the reasons for bringing these people and artifacts together may vary, all of these theories argue for repetition and temporal stability in this organization. Arguably, the convergence of many of these types of stable organizations of people and artifacts constitutes the predictable, and hence intelligible, aspects of context [11].

In addition to the philosophical argumentation of these theories, they also provide more tangible benefits [23] – specifically descriptive power. The best approach to analyzing the descriptive power of a theory is to ground its use in real data. Many studies have done so, e.g. [5, 9, 12, 13, 15, 17, 26]. These studies essentially confirm that the structures mentioned above are observable in information workers' day-to-day behavior. Thus, to capture some of the context of an information worker's multi-tasked workflow, a system needs to generically handle many types of context structures.

These theories can be distilled and captured by two generic features of context structures.

1. Context structures are repeated in a relatively stable way through the routine practices of people.

2. Context structures evolve as people's routines evolve.

In the next section, we discuss how these patterns are found, represented, and used by CAAD.

SYSTEM DESCRIPTION

CAAD's design objective is to support computer-based information workers with a minimum amount of interaction overhead. To meet this objective, CAAD coordinates three components: (1) a logging component that captures computer interaction events, (2) a pattern mining component that performs offline and online calculations related to the user's evolving context structures, and (3) an awareness component that displays the context structures in a direct manipulation UI. Figure 1 illustrates these components. We discuss each component in turn.

Logging Component

The logging component gathers evidence of information use on the computer. This evidence consists of interaction events like file access and modification, email transmission, application use and state, and web browsing activity. Many related systems capture similar events [16, 27, 28]. Most of these systems target Microsoft Windows users and rely on various hooks into the input stream or the COM interface to capture events. They are "push" architectures – events are pushed at the system. Alternatively, our logging component uses a "pull" architecture. It periodically checks for relevant events and state changes.

The decision to pull information stems from the type of events we are interested in. All of the events that are logged can be described as "using X" where X can be applications, files, folders, web pages, and email addresses. If our logger was only receiving push events like "file X was opened" and "file X was closed", it would need to maintain state variables to determine "using file X". These state variables would be sensitive to missed events, requiring potentially sophisticated back-up mechanisms.

Files that are logged can be of many origins and types. Specifically, files on local and network drives, web pages, email attachments, email subject lines (as short text documents), and email body texts are logged. Additionally, file use produces two events: one with the file path name and another with an md5 hash of the file contents. By creating two events, we can track changes on a single file (same file path name, different md5 hashes) as well as file moves (different file path name, same md5 hash). Email-related events are restricted to outgoing email because incoming email has limited correlation to work-flow at the second or minute time-scales. (We originally included incoming email, but the resulting patterns were often inaccurate due to "out-of-context" messages.)

Application use events are logged using several redundant pieces of information. First, the logger tracks active windows. This is necessary because many computer users leave applications and windows open, even if they are not being used. The list of active windows generates a list of active applications that time-out after 30 seconds – i.e. if an application has not been the active window in the last 30 seconds it is no longer considered active. This list is cross-checked against the list of running applications. Applications that are both active and still running are logged. Finally, if the active window has not changed in five minutes, the logger assumes the user is taking a break and no applications are logged.

As described earlier, context structures are sets of relevant tools (in this case files, folders, web pages, and applications) and people (referenced by email addresses). To accurately infer these structures, CAAD must know when, and for how long, tools have been used and people have been communicated with. In line with this, the logging component polls for events once every 2 seconds[1], depending on CPU load. The logging component itself averages to about 5% CPU load on a Pentium 4 with 512mb of RAM (spiking during the calculation of file hashes).

Pattern Mining Component
The pattern mining component detects generic context structures (see the Conceptual Background section) in logs of computer interaction events. It consists of a pattern detection algorithm and the necessary functions for pre- and post-processing of the data and results. The algorithm is a variant of GaP [4]. Basically, it performs a dimensionality reduction calculation, similar to Latent Semantic Analysis or Principal Component Analysis. It differs from all of these algorithms by also tracking the slow evolution of context structures in a principled and intuitive way. A detailed discussion of the algorithm is beyond the scope of this pa-

---

[1] Information access events at shorter time scales are missed. However, preliminary experiments did not reveal any events that would justify polling at a higher rate.
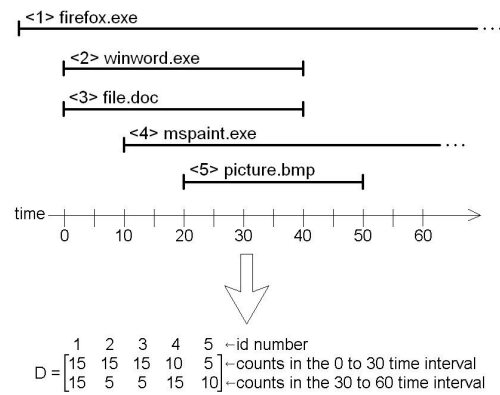
Figure 2. Illustration of how usage events are converted into the input matrix, D, for the pattern mining algorithm. (Polling happens about once every 2 seconds.)

per (a full description is currently in preparation). However, we cover pertinent details in separate sub-sections below.

Input to the Data Mining Algorithm
Logged events are grouped into contiguous segments of time. Currently these segments are 30 seconds long (we have experimented with a number of different segment lengths, as well as combinations of varying length segments, but the results did not vary significantly). Within each segment, CAAD maintains counts over events. For example, if Kate, the knowledge worker, has a document open for an entire 30 second interval and the logger is polling once every 2 seconds, then the system will have 15 use events for the document. If the document is a PDF, there will also be 15 use events for Acrobat. Additionally, if Kate also looked at another document (in Microsoft Word format) for 10 seconds during the same interval, the logger would record 5 usage events for this document during the interval as well as 5 usage events for Word. Events that did not occur in that time window will have a count of zero. Aggregating all of these time segments together, the input to the pattern mining algorithm is a large non-negative matrix with integer elements, most of which are zero. Figure 2 illustrates this input process.

Offline vs. Online Context Structure Calculations
CAAD's model of the user's context structures is stored in a non-negative matrix. Each context structure corresponds to a row of this matrix, and each entry in the row corresponds to the probability of observing a specific usage event provided the user is working on the context structure associated with that row.

Context structures are calculated *offline* – once per day – with the most recent 4 weeks of logged event data. On average, with 4 weeks of event data, the algorithm takes between 10 and 20 minutes to run an offline update on a 2 GHz Pentium 4 with 512mb of RAM. CAAD is currently configured to perform this update in the middle of the night. However, it is reasonable to run the update during a lunch break or a meeting if overnight updates are not feasible.
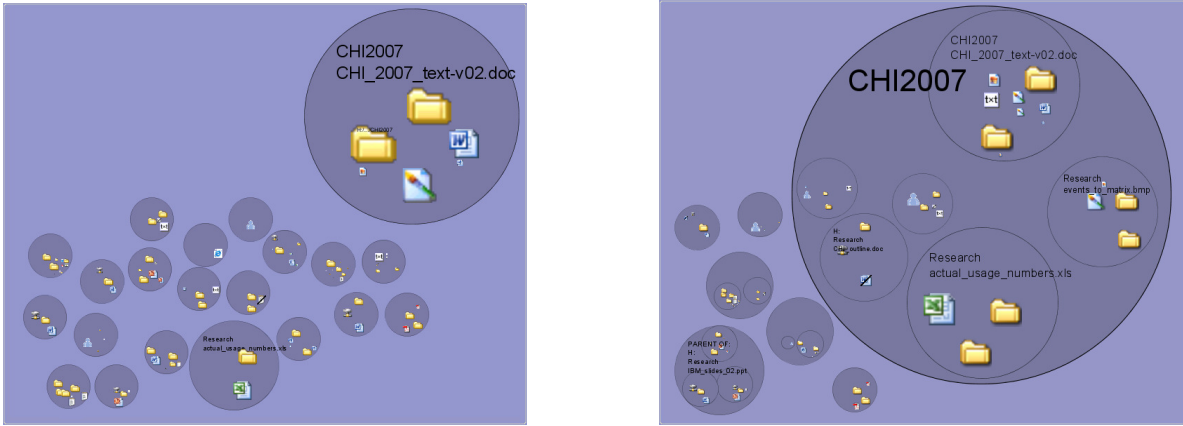
Figure 3. Two screen-shots of the activity display. The left image was automatically generated by CAAD. The right image illustrates a user-edited display. Information items are organized into groups corresponding to the user's context structures. The sizes of the groups are dynamically set (every 30 seconds) so that more relevant groups and information items are larger – making them the easier to access. New groups appear to the right and more frequently used groups appear towards the top.

An important characteristic of the offline calculation is that it tracks the evolution of context structures. This is important because people's activities change and evolve [18], requiring any task support system to handle these changes. CAAD handles these changes by biasing the latest calculation using the results of the previous calculation. Intuitively, this bias results in a time-shifting average of the user's context structures. The amount of bias is set so that context structures undergo minor changes like the adding or removing of one or two relevant events per day. At this rate, most context structures can completely change (i.e. total replacement of the relevant events) about once every 4 weeks.

CAAD's model of the user's context structures remains fixed between daily updates. Between daily updates, it is used to make *online*, i.e. real-time, predictions on what the user is doing – and hence what information is relevant to him or her. The online updates basically calculate how likely the user is to be working on each context structure. Online updates require minimal memory and computational resources – they are negligible in comparison to the requirements of the logging component.

Determining the Number of Context Structures
Unlike related dimensionality reduction algorithms, the pattern mining component in CAAD automatically determines the number of context structures (i.e. dimensions) to calculate. It does this using a simple greedy search. As the algorithm runs, it checks whether two context structures cover a similar set of time intervals and whether they use a similar set of events. If two structures are too similar, determined by a threshold that we hand-tuned, they are merged. This process is continued until the set of calculated context structures has stabilized numerically.

Awareness Component
The awareness component of CAAD is a display showing the user's activity, represented as a set of context structures (which are detected by the other components of CAAD). The display is dynamically configured according to online,

i.e. real-time, predictions on which context structures are relevant to the user. Predictions are made every 30 seconds. The display is the only component of CAAD that the user directly interacts with.

The display supports users in two ways. First, it acts as a portal through which users can access information relevant to their various projects and tasks. Second, it provides a mechanism for users to become more reflective about the organization of their work behavior. The display provides this support in a context-aware way be leveraging the real-time predictions of what is relevant to the user. The predicted relevance of an element determines its size in the display – the most relevant information items are the largest elements in the display and hence easiest to access.

The information in the display, as discussed above, represents the implicit context of the user's work-flow. By providing this information, the activity display enables users to be more reflective on how they organize their day-to-day work routines. This reflection could be superficial (e.g. answering "What have I been working on lately?") or more profound (e.g. realizing that "I look at sports news web pages every few minutes. Maybe I should try restrict these breaks and see if I am more productive or not."). Although the activity display can support these types of reflection, it does require some reflective inference on the part of the user – for example observing that there are sports news web pages in every structure found by the system and interpreting what this means about how they organize their work-flow.

Figure 3 shows two screen-shots: one from a default configured (i.e. no edits have been performed) activity display (left), and one from a user-edited display (right). The context structures are represented as circular nodes in the display. For convenience, we will refer to the context structure nodes as *groups*. Each group contains icons representing relevant files, folders, web pages and people. We will refer to these icons, and the things they represent, as *information items*. *Parent* nodes, which are containers for groups, are

only shown in Figure 3 right. Currently, parents are not automatically generated; users must manually create them.

By default the display only shows information items and groups. The groups correspond to the context structures found by the pattern mining component of CAAD – updated once per day during the offline calculation. In the display, only the most relevant information items are shown. The threshold on which information items to show is determined by the most relevant information item for that structure. Any items with a relevance weight greater than 5% of the weight of the most relevant information item are shown. In addition to this threshold-based scheme, certain types of information items are systematically not considered for display. These items include hash values of file content and applications. Applications are not included in the display because each application is often common to many groups and hence provides little descriptive value. Furthermore, applications can be accessed indirectly by opening information items. (Part-way through the study we added a menu selection item allowing applications to be viewed in the display if the user chooses. Only one participant exercised this option.)

As with information items, not every group (i.e. context structure) is shown in the display. Some groups, which consist primarily of non-displayable information items, are hidden by default. For example, a group which consisted entirely of applications would not be shown. For all of the users in our study, between 70% and 80% of their context structures were displayed – corresponding to a range in number of between 3 and 37 groups.

In addition to the visual representations, every element in the display has a textual label. For information items, the labels are an abbreviation of the path, URL, or filename. To create group labels, the top five labels from all the contained information items, according to relevance weight, are concatenated.

The display supports the following user interactions:

- navigation through the display using mouse clicks,
- accessing information items by double-clicking,
- listing the contents of any group or parent element,
- adding/removing elements,
- changing the label text of elements, and
- changing the relevance of information items within groups or groups within parents.

Structural edits (e.g. relevance changes, additions, removals) performed by the user modify CAAD's representation of their context structures. These edits can be performed either through context menus or via direct manipulation – by dragging elements into and out of one another.

Discussion of System

A constant concern with CAAD, and generally any logging system, is privacy. Although all of the events that CAAD logs are already collected by Windows or other applications, there might be some additional risk related to the aggregation of these events. We currently address this concern by (1) storing the log events in a single location, which the user of the system can easily access and delete; and (2) performing all of the necessary calculations on the data locally. For future applications of CAAD that require some sharing of detected context structures, we are planning on refactoring the pattern mining component so that calculations can be encrypted and run over a centralized network of computers [8].

A practical concern with the current logging architecture is that it can require significant computational resources if, for example, there are many files whose contents needs to be hashed or if the history files it reads become large. One participant in our study complained about degraded system performance. However, we found that this performance was the result of poor memory performance, only partially attributable to CAAD.

Finally, the display in CAAD must balance the natural tension between showing updated structures and retaining enough visual cues between updates so that the user can utilize recognition rather than search and recall. We handle this balance by letting the system generate a new layout after each update, and then modifying this new layout using the previous layout details – e.g. labels from the previous layout take precedence; any information items that were previously but are no longer in a group are added to the new layout with a discounted relevancy weight; and any new information items in a group are shuffled spatially so that previously included information items can be shown in their original positions. The differences between the new and previous layouts can be highlighted using a special color scheme (which the user can toggle on and off from the context menu). The axes are also ordered using clear semantics. The x-axis corresponds to time – newer context structures are further to the right. The y-axis corresponds to the total amount of time the user spent on each context structure – more time moves the context structure up.

In the next section, we describe the study we ran to verify that CAAD could, and does, support information workers.

USER STUDY

The overall objective of this was to demonstrate the feasibility of our approach to supporting computer-based information workers by automatically inferring the context structures of their day-to-day work behavior.

The study started with 10 participants (7 graduate student, 3 undergraduate students) and was conducted in the actual work settings of each participant. All of the participants were working on single-monitor, desktop computers prior to the study. For the study they were provided with a sec-

Perceived usefulness:	mean	std. dev.
Using the activity display while I work would enable me to accomplish tasks more quickly.	1.63	0.52
Using the activity display would improve my work performance.	0.63	0.92
Using the activity display would make it easier to do my job.	1.25	0.71
Using the activity display would enhance my effectiveness at work.	0.88	0.83
Using the activity display while I work would increase my productivity.	0.88	0.83
I would find the activity display useful in my work.	1.00	0.93
Average perceived usefulness score	1.04	0.69

Perceived ease-of-use:	mean	std. dev.
I find it easy to get the activity display to do what I want it to do.	1.13	0.99
My interaction with the activity display is clear and understandable.	1.25	1.04
Learning to operate the activity display was/is easy for me.	1.25	1.83
It was/would be easy for me to become skill-ful at using the activity display.	2.00	0.82
I find the activity display easy to use.	1.50	0.93
Average perceived ease-of-use score:	1.42	0.92

Figure 4. Questionnaire results. Questions were scored on a 7-point Likert scale ranging from -3 to +3. Mean response values and standard deviations are reported for each question and for the averaged response values.

ond monitor, used primarily, but not exclusively, for the activity display. Usage was not strictly enforced and participants often placed other application windows over the activity display to maximize their screen space.

The key characteristic for all participants was their involvement in some form of computer-based information work. To assess this, we administered an initial questionnaire that asked participants about the amount of time they spent on the computer and what tasks or projects they worked on during this time. Example projects found in this questionnaire include: preparing lecture notes, managing a small research team, conducting studies, processing study data, searching for and reading related papers, and developing software. The vital concern that the initial questionnaire was designed to assess is whether the participants spent a significant amount of time working on their computers. If they did not, then detecting their computer-based context structures would be of limited value in supporting their work. Based on the questionnaire results, we were confident that the participants in the study could be supported by CAAD.

During the study we collected data from three sources: questionnaire results measuring perceived usefulness and ease-of-use, logs of actual usage events with CAAD's display, and semi-structured interviews. Before describing the results we discuss the method details for each of these data sources.

Methods

Perceived Usefulness and Ease-of-use Questionnaire
The questionnaire we used measures perceived usefulness and perceived ease-of-use [6] (Figure 4 contains the actual questions we used). All questions were scored on a 7-point Likert scale, ranging from -3 to +3. Scores greater than zero indicate that the participant found the system useful (or easy to use). We modified the standard questionnaire (prior to administering it) by removing the lowest correlating question for ease-of-use. This question targeted system flexibility, which is not always correlated with ease-of-use or with overall system usage [6].

Only 8 of the original 10 participants completed the questionnaire after the first week of the study – system compatibility issues with the other two participants resulted in negligible exposure to CAAD during the first week of the study. Additionally, we had 5 of the 8 participants take the questionnaire a second time later in the study – the other three participants were unavailable for subsequent interviewing. The second application of the questionnaire was designed to capture two things. The first was to assess novelty effects in the first application of the questionnaire; and the second was to assess the effects of longer-term use of CAAD.

Actual Usage of CAAD's Display
To assess how well CAAD met its design goal of supporting access to information we logged usage events with the display. These logs include every information access as well as every edit or modification (Figure 5 shows the events that were logged). We collected these logs for 7 of the original 10 participants during the first week of the study – system compatibility issues resulted in minimal exposure to CAAD for two participants and a third participant could view the display but not interact with it.

Interviews
In addition to validating the questionnaire results, interviews were used to verify the accuracy of the context structures found by the pattern mining component. We interviewed 8 of the original 10 participants – excluding the two with minimal exposure to CAAD due to system compatibility issues. The interviews were semi-structured. They started with general questions about the participants' experience with CAAD and then walked through most of the groups in the activity display. We were particularly interested in specific examples of groups that were or were not accurate in the participant's opinion. Through these examples, we hoped to assess whether the participants understood the underlying algorithm, what their expectations were for its behavior, in what ways they actually found the display useful, and if they had any interaction issues or any privacy concerns

Results

Perceived Usefulness and Ease-of-use Questionnaire

Again, all questions were scored on a 7 point Likert scale, ranging from -3 to +3. Both the average perceived usefulness responses (T-stat = 4.2785, df = 7, p <= 0.0037) and the average perceived ease-of-use responses (T-stat = 4.36, df = 7, p <= 0.0033) were statistically significance away from 0 (a neutral question response) in the positive direction. This means study participants found the activity display both useful and easy to use. Specific question results are shown in Figure 4.

Of the five participants that took the questionnaire a second time, three took it after the second week of the study, one after the third week of the study, and one after the fourth week of the study. The averaged score differences were all positive (indicating an increase in perceived usefulness and perceived ease-of-use) with the exception of one person on ease-of-use (this participant felt like the semantic zooming – which hides labels when they are too small – made the display hard to quickly read). The overall increase in questionnaire scores indicates that: (1) the initial questionnaire results were not artificially inflated due to novelty affects, and (2) that CAAD was able to effectively track the evolution of people's context structures over multiple weeks.

Actual Usage of CAAD's Display

Actual usage results are presented in Figure 5. Aggregating the seven usage logs, we found that 36% of the usage events were information access events. The majority of edit events corresponded to the deletion of an information item from a group. On average, users generated 8.1 events per day (the logs covered at most three days worth of interaction for each participant). However, most participants deviated from this average significantly. The least active participant only generated 7 total events while the most active generated over 50. The variance in these usage results, coupled with the questionnaire results, provides some evidence of CAAD's ability to support information workers with different working styles.

We also calculated a derived metric of CAAD's utility from the usage logs. If we count the number of elements displayed by CAAD and divide by the number of structural edits that users made, we get an estimate of the value CAAD adds relative to a manual task-management system (where users have to insert and group information items from scratch). The validity of this metric is conditional on the perceived usefulness of CAAD – basically we assume users performed as many edits as were required to develop the opinion that CAAD is useful. Likely, however, users performed more edits than this, making this metric a conservative measure of CAAD's utility.

Values greater than 1.0 for this metric indicate benefit to the user. For the seven participants that we have usage data for, these values ranged from 1.5 to 68.0 (mean 18.4, std dev 25.2, not statistically significant above 1.0). The fact that all of these values are greater than 1.0 indicates that

Usage events per participant

								Totals
access event								61
delete element								51
add info. item								18
label change								12
merge groups								10
add parent								8
subtract group								7
add group								2
change relevance								1
	25	18	20	23	7	19	58	Totals

Figure 5. Visualization highlighting actual usage patterns. The seven middle columns correspond to participants. Shading linearly scales with the percentage of that type of event, per participant. Column totals are the number of events generated per participant. Row totals are the number of each specific event type generated by all participants.

CAAD provides an overhead reduction relative to manual task management systems.

An alternatively way to measure the utility is to take the difference instead of the ratio. Whereas the ratio metric can be thought of as a relative measure of utility (relative to how many edits each user made), the difference based metric captures more of an absolute measure of utility (how many edits did the user save). Values for the difference based metric ranged from 6 to 177 (mean 77.3, std dev 61.4) and were statistically significant above zero: T-stat = 3.3291, df = 6, p <= 0.016. Again, the fact that all of these values are greater than zero indicates that CAAD provides an overhead reduction relative to manual task management systems.

Interviews

To start the interview, we asked participants to describe their overall impressions of CAAD. The comments we received were evenly split between positive accolades (e.g. "it makes reasonable prediction on what things belong together") and more negative criticisms (e.g. "groups are mostly correct – sometimes they have a few additional things that don't belong"). In line with the goal of determining the feasibility of supporting information workers by automatically detecting their context structures, the remainder of the interviews focused on eliciting and understanding negative and/or critical comments.

However, in pursuing the evidence that participants had for their critical comments, we found that they were the result of contradictions between the temporal correlation patterns captured by CAAD and the mental models participants had of their own work. The critical comments uncovered in our interviews revealed four common contradictions.

The most frequent comment was that groups contained information items that did not belong – e.g. web pages related

to sports or news in groups containing mostly work documents. About 40% of groups across the eight participants we interviewed were subject to this comment. The notion of belonging was determined by the participants' mental models of their work – e.g. knowing a sports score does not help them complete their work. However, they were aware of reading sports and news pages while they worked. These breaks, or rather micro-breaks, generally lasted less than one minute and occurred at most a few times per hour. Because these breaks were so short, participants often left the documents and applications relevant to their current task open. Hence, in terms of temporal correlation, these sports and news web pages seem to belong in the groups associated with their task. They are, in a counter-intuitive sense, part of the context of the participant's routine work-flow. With more elaborate content analysis, it might be possible to separate them, but we think for now it is more appropriate to include them in the activity analysis. At the very least they may be indicators of context switches and trigger associations between the users' main tasks.

Another frequent comment was that some groups were related to one another and should have been merged – either into a single group or as children of a single parent (we estimate about 25% of groups were subject to this comment). For example, one participant said: "My 'preparing lecture notes group' really belongs with my 'preparing the midterm group'." However, with additional questioning, we found that the participant had not worked on these tasks at the same time, nor did they share many information items. Thus, in terms of working context, these groups were practically disjoint. Some participants dealt with this problem by creating parent nodes and grouping context structure nodes together (Figure 3 illustrates this).

A third comment, made by three participants, was that there were too many groups in the display. One participant even stated that no matter what he was working on, he did not want to see more than 4 or 5 groups in the display. However, while discussing the groups in the display, only two of seventeen were not readily identifiable by him. In other words, this participant knew he worked on more than 5 things but did not want to be shown these things by CAAD. The other two participants who made this comment had similar, although less extreme, sentiments.

Finally, some participants made comments about groups missing relevant documents (we estimate about 25% of groups were subject to this comment). Like the group merging comment discussed above, these comments highlighted an abstract connection in the participant's mental model of their work that was not part of their day-to-day work routine. With a few extra questions, it was clear that they had never used the "missing" document while working with the other information items in the group – i.e. there was no temporal coordination between their uses. Rarely, some missing documents were never actually logged by CAAD. We discuss this problem in the next section.

DISCUSSION

Most notably, based on the questionnaire results, people found CAAD useful and easy to use. Using a metric derived from the amount of task structure automatically detected by CAAD and the number of edits people performed on these structures, we showed that CAAD provides clear overhead reduction relative to manual task management approaches. We also suspect that CAAD requires less overhead than semi-automatic methods, although we have not specifically studied this difference.

There are a number of interesting caveats and subtleties to CAAD worth mentioning. First, CAAD separates potentially related context structures if they are not temporally correlated. For example, two groups representing different phases of the same project will only be represented as a single group (i.e. context structure) if the user works on both of them at the same time.

Second, one participant made the comment that "[CAAD] definitely shows relevant things. But they are not always useful." This participant was referring to his lecture preparation for the course he was teaching. While preparing new slides, CAAD would show him slides from previous lectures he had already written. Although related, these old slides were not really useful in preparing the new slides. CAAD, in its current implementation, can only support an ongoing task that re-uses the same (or a similar) set of documents and people. CAAD will not be able to track a context structure if the relevant documents and people change too quickly.

Third, some applications (e.g. Emacs) neither make a native OS call nor create a lock when opening a file. Consequently, they are invisible to the CAAD. They do, however, typically create a temporary file. We are exploring extensions to the logging component to capture these events; but, in the meantime, we will screen for participants that only use applications and files currently supported by CAAD.

Fourth, the disparity between user's mental model of their work and the context structures that CAAD uncovers offer some interesting design issues. Generically, the contradictions revealed in our interviews fall into two classes: (1) those where the user's mental model contained a connection between disjoint, temporally uncorrelated context structures and (2) those where the context structure contained connections that were not part of the user's mental model. Although making the displayed content of CAAD easily comprehensible is a natural imperative, it may be an impossible ideal. Interestingly, it may be more appropriate to make the display playful and mysterious by including and possibly highlighting the contradictions, thereby inviting the user to explore CAAD's perspective on their work behavior. An interesting study could measure the relative frequency of these contradictions and whether one or the other lead to more beneficial improvements for the user when they became aware of it. This and related issues are left as future work.

CONCLUSION AND FUTURE WORK

In this paper, we presented CAAD – a context sensitive system that supports information workers by improving their access to relevant information and by providing reflective feedback to improve task management. This support is achieved with minimal overhead because CAAD functions automatically. With sufficient iteration on its design, we believe CAAD could become a common task management tool for information workers.

We conclude with several directions for future work directions. First, we could improve the logging component of CAAD to capture applications and files that are currently missed. Second, we could iterate the display design to improve the animation capturing the real-time predictions (which three participants found distracting). Third, we could explore methods for automatically grouping semantically related context structures together (e.g. different aspects of the same project).

More significantly, we can use the infrastructure of CAAD (the logging and pattern mining components) to build new applications. Some ideas we are currently exploring include promoting information sharing among group members based on task similarity and improving information retrieval by task-specific query augmentation. We could also extend the pattern mining component of CAAD to search for multi-person context structures or move CAAD into new settings like smart rooms or onto devices like cell phones or PDAs.

REFERENCES

1. Bakhtin, M. Speech genres and other late essays, (1979).

2. Bardram, J., Bunde-Pedersen, J., and Soegaard, M. Support for activity-based computing in a personal computing operating system. Proc. of SIGCHI, (2006).

3. Bellotti, V., et. al. What a to-do: studies of task management towards the design of a personal task list manager. Proc. of SIGCHI, ACM (2004), 735-742.

4. Canny, J. GaP: a factor model for discrete data. Proc. of SIGIR, ACM (2004), 122 - 129.

5. Czerwinski, M., Horvitz, E., and Wilhite, S. A diary study of task switching and interruptions. Proc. of SIGCHI, ACM (2004), 175-182.

6. Davis, F. Perceived Usefulness, Perceived Ease of Use, and User Acceptance of Information Technology. MISQ 13(3), (1989), 319-340.

7. Dourish, P. What we talk about when we talk about context. Personal Ubiquitous Computing 8(1), (2004).

8. Duan, Y. and Canny, J. Zero-knowledge Test of Vector Equivalence and Granulation of User Data with Privacy. Proc. of Granular Computing, IEEE (2006).

9. Engeström, Y., Engeström, R., and Vähäaho, T. When the center does not hold: the importance of knotworking. Activity Theory and Social Practice, (1999), 345-374.

10. Fogarty, J., et. al. Predicting human interruptibility with sensors. ToCHI Journal 12(1), ACM (2005), 119-146.

11. Giddens, A. The Constitution of Society, (1984).

12. Goffman, E. Frame Analysis: an essay on the organization of experience. Northeastern Univ. Press (1974).

13. González, V. and Mark, G. Managing currents of work: multi-tasking among multiple collaborations. Proc. of ECSCW, (2005).

14. Henderson, D. and Card, S. Rooms: the use of multiple virtual workspaces to reduce space contention in a window-based graphical user interface. Transactions on Graphics Journal 5(3), ACM (1986), 211-243.

15. Hutchins, E. Cognition in the wild. MIT Press (1995).

16. Kaptelinin, V. UMEA: translating interaction histories into project contexts. Proc. of SIGCHI, (2003), 353-360.

17. Law, J. Notes on the theory of the actor-network: ordering, strategy and heterogeneity. Systems Practice 5(4), (1992), 379-393.

18. Leontiev, A. Activity, consciousness, and personality. http://lchc.ucsd.edu/MCA/Paper/leontev/, (1978).

19. Moran, T. Unified Activity Management: Explicitly Representing Activity in Work-Support Systems. Workshop paper at ECSCW, (2005).

20. Morteo, R., et. al. Sphere Juggler: fast context retrieval in support of working spheres. Proc. of Mexican Conference in Computer Science, (2004).

21. Muller, M., et. al. One Hundred Days in an Activity-Centric Collaboration Environment Based on Shared Objects. Proc. of SIGCHI, ACM (2004).

22. Nardi, B. (editor). Context and Consciousness: Activity Theory and Human-Computer Interaction, (1996).

23. Nardi, B. and Redmiles, D. (editors). JCSCW special issue: Activity Theory and the Practice of Design 11(1-2), Kluwer (2002).

24. Robertson, G., et. al. The task gallery: a 3D window manager. Proc. of SIGCHI, ACM (2000), 494-501.

25. Smith, G., et. al. GroupBar: The TaskBar Evolved. Proc. of OZCHI, (2003).

26. Spinuzzi, C. Tracing genres through organizations: a sociocultural approach to information design, (2003).

27. Stumpf, S., et. al. The TaskTracer system. Proc. of AAAI, (2005).

28. Voida, S., et. al. Integrating virtual and physical context to support knowledge workers. Pervasive Computing 1(3), IEEE (2002), 73-79.

29. Wasserman, S. and Faust, K. Social Network Analysis: Methods and Applications, (1994).