

Task Analysis for Groupware Usability Evaluation: Modeling Shared-Workspace Tasks with the Mechanics of Collaboration

DAVID PINELLE and CARL GUTWIN

University of Saskatchewan

SAUL GREENBERG

University of Calgary

Researchers in Computer Supported Cooperative Work have recently developed discount evaluation methods for shared-workspace groupware. Most discount methods rely on some understanding of the context in which the groupware systems will be used, which means that evaluators need to model the tasks that groups will perform. However, existing task analysis schemes are not well suited to the needs of groupware evaluation: they either do not deal with collaboration issues, do not use an appropriate level of analysis for concrete assessment of usability in interfaces, or do not adequately represent the variability inherent in group work. To fill this gap, we have developed a new modeling technique called Collaboration Usability Analysis. CUA focuses on the teamwork that goes on in a group task rather than the taskwork. To enable closer links between the task representation and the groupware interface, CUA grounds each collaborative action in a set of group work primitives called the *mechanics of collaboration*. To represent the range of ways that a group task can be carried out, CUA allows variable paths through the execution of a task, and allows alternate paths and optional tasks to be modeled. CUA's main contribution is to provide evaluators with a framework in which they can simulate the realistic use of a groupware system and identify usability problems that are caused by the groupware interface.

Categories and Subject Descriptors: D.2.2 [**Software Engineering**]: Design Tools and Techniques—*User interfaces*; H.5.2 [**Information Interfaces and Presentation**]: User Interfaces—*Graphical user interfaces, User-centered design, Evaluation/methodology*; H.5.3 [**Information Interfaces and Presentation**]: Group and Organization Interfaces—*Computer-supported cooperative work, Evaluation/methodology*

General Terms: Design, Experimentation, Human Factors

Additional Key Words and Phrases: Groupware evaluation, Groupware usability, Mechanics of collaboration, Task analysis

This research was supported by the National Institute for Standards and Technology and by the Natural Sciences and Engineering Research Council of Canada.

Authors' addresses: David Pinelle and Carl Gutwin, Department of Computer Science, University of Saskatchewan, Saskatoon SK, S7N 5A9; email: {david.pinelle,carl.gutwin}@usask.ca; Saul Greenberg, Department of Computer Science, University of Calgary, Calgary AB, T2N 1N4; email: saul@cpsc.ucalgary.ca.

Permission to make digital or hard copies of part or all of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or direct commercial advantage and that copies show this notice on the first page or initial screen of a display along with the full citation. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, to republish, to post on servers, to redistribute to lists, or to use any component of this work in other works requires prior specific permission and/or a fee. Permissions may be requested from Publications Dept., ACM, Inc., 1515 Broadway, New York, NY 10036 USA, fax: +1 (212) 869-0481, or permissions@acm.org.

© 2003 ACM 1073-0616/03/1200-0281 \$5.00

1. INTRODUCTION¹

Poor usability in many distributed shared-workspace systems has prompted CSCW researchers, including ourselves, to develop discount usability methods for designing and evaluating groupware [Pinelle and Gutwin 2002; Steves et al. 2001; Greenberg et al. 1999; Cugini et al. 1997]. Discount methods cost less than traditional groupware evaluation techniques such as field studies or controlled experiments, and focus more closely on interface usability issues than on adoption or patterns of use in organizations. Discount methods work well with low fidelity prototypes, which allows evaluations to take place during early development when there is no operational prototype for users to test in a real work setting.

These discount methods require some means of understanding and representing the tasks that groups will perform; since evaluation does not happen within the real work setting, information about the task context must be articulated and used synthetically. Consider the most popular approaches: scenario-based walkthroughs demand well-articulated and realistic tasks so that inspectors can walk through them step by step while driving the interface; similarly, heuristic evaluation recommends that inspectors keep a task context in mind while checking to see how the interface complies with guidelines; finally, good usability studies demand that evaluators give subjects realistic tasks if they are to observe true problems and successes with the interface. Yet there are currently no modeling or analysis schemes appropriate for groupware usability evaluation. The problem is that existing task modeling schemes are either unable to represent the flexibility and variability that is inherent in real-world group task behaviour, or use a level of analysis that is too broad to allow for usability evaluation of a group interface.

To address these limitations, we have developed a new modeling technique called *Collaboration Usability Analysis* (CUA). CUA incorporates several modifications to other task analysis techniques in order to make it more appropriate for developing and evaluating groupware systems. In particular, CUA allows variable paths through the execution of a task, allows alternate paths and optional tasks to be modeled, and specifies teamwork with a set of low-level operations called *the mechanics of collaboration*. CUA allows designers to quickly model the main features of a group work situation that will affect groupware usability, and to use these models in discount evaluations such as inspections and walkthroughs. CUA helps development teams iteratively develop and evaluate multi-user systems, thus improving the usability of shared workspaces.

In this article, we introduce Collaboration Usability Analysis and describe its structures and uses. We begin by reviewing the motivating problems faced by groupware developers, and identify the limitations that we see in existing task analysis techniques. We then describe the details of CUA, and provide examples

¹This article contains concepts and material from previous conference and workshop papers, but included material has undergone substantial revision, and the main focus of the article is new unpublished work.

of how we have used CUA to build task models of collaborative activities and to evaluate groupware systems.

2. MOTIVATION: GROUPWARE DEVELOPMENT AND EVALUATION

In user-centered software engineering, developers rapidly iterate through a process of design, implementation, and evaluation. Much of this iterative development is focused on the early detection and repair of usability problems. To be practical, it should be possible to carry out each step in the development cycle quickly, easily, and at reasonable cost. For maximum impact, we need evaluation techniques that work with early low-fidelity interface designs. This is the reason why discount evaluation methods have been so well accepted, as they help evaluators rapidly find usability problems even in very early system prototypes. Popular examples include consistency inspections and standards inspections [Wixon et al. 1994], pluralistic walkthroughs [Bias 1994; 1991], cognitive walkthroughs [Lewis et al. 1990; Polson et al. 1992], and heuristic evaluations [Nielsen and Mack 1994; Nielsen and Molich 1990]. Although discount methods have limitations (such as dissociation from the real work settings and a lack of a real theoretical basis), they have proven to be valuable tools in software development. Even without actual users in the actual work setting, these methods have become successful because they provide evaluators with enough detail about the work and task context for them to find legitimate usability problems.

Although developing groupware interfaces is similar in many ways to developing interfaces for traditional single-user applications, there is a crucial difference—groupware developers cannot yet rapidly iterate through the design, implementation, and evaluation process. Traditional discount evaluation techniques do not work well for evaluating groupware, which means that early groupware prototypes cannot be effectively evaluated. The main problem is that discount evaluation methods are strongly oriented around individual work: the contextual information they provide and the criteria they use for judging usability are focused on tasks and actions that individuals must carry out in working towards a goal.

Recently, we and others have devised new discount methods that can assess groupware usability [Pinelle and Gutwin 2002; Steves et al. 2001; Baker et al. 2002; Cugini et al. 1997]. These methods view group activity as being divided up into two areas: *taskwork*, the actions needed to complete the task, and *teamwork*, the actions needed to complete the task as a group—“the work of working together.” Where singleware usability evaluations assess support for taskwork, groupware usability evaluations must assess support for teamwork. This becomes our definition of groupware usability: the extent to which a groupware system allows teamwork to occur—effectively, efficiently, and satisfactorily—for a particular group and a particular group activity. While still early work, groupware evaluation techniques now include: basic inspection [Steves et al. 2001], a walkthrough technique based on cognitive walkthrough [Pinelle and Gutwin 2002], and a variant of heuristic evaluation [Baker et al. 2002].

These techniques show considerable promise for filling a gap in the groupware designer’s toolbox. However, one remaining concern is how teamwork can

be modeled for use in these usability evaluations, particularly since certain techniques such as pluralistic walkthroughs and cognitive walkthroughs require a fairly detailed task model. Although there are task analysis schemes in existence for both single-user tasks [Diaper 1989; Richardson et al. 1998] and multi-user situations [van der Veer et al. 2000, 1997; Paternò et al. 2001], existing schemes are not well suited to modeling teamwork for discount groupware evaluation. Based on our experience with groupware evaluation techniques, we identified two additional requirements that must be addressed in a task analysis scheme that is appropriate for modeling teamwork.

1. *Help evaluators identify teamwork usability problems that are caused by the groupware interface.* The scheme must be able to represent the details of collaborative interaction such that those actions can be mapped to specific components and structures in the groupware system.
2. *Help evaluators understand and explore the range of ways that the groupware system will be used* (and thus find a wider set of usability problems). The scheme must be able to represent the variability in teamwork and taskwork that is inherent in group activity.

We have addressed these two additional requirements in CUA, which is introduced and detailed in the following sections.

3. AN OVERVIEW OF COLLABORATION USABILITY ANALYSIS

Collaboration Usability Analysis is a task analysis technique designed to represent collaboration in shared tasks for the purpose of carrying out usability evaluations of groupware systems. CUA is focused on the teamwork aspects of a collaborative situation: it provides both high-level and low-level representations of the collaborative situation and the shared task, and provides ways to represent multiple actors and the interactions between them in shared work. While CUA provides task model components to represent individual work, we are primarily interested in the teamwork aspects of group work since this is what sets groupware apart from other applications.

CUA is based on a hierarchical task model that represents the procedural elements of a group task in a shared workspace. Several basic features of the scheme are similar to hierarchical task analysis [Annett and Duncan 1967; Shepherd 1989], a method that provides flexibility in the ways that tasks are composed and executed. In addition, group features such as actors and roles are similar to the Groupware Task Analysis (GTA) method [van der Veer et al. 2000, 1997]. CUA also handles some of the complexities that are seen in ConcurTaskTrees [Paternò et al. 1997], such as variation in task execution, and iterative and optional tasks. Unlike these other approaches, however, our scheme is oriented around collaboration, and uses a lower and more concrete level of analysis. In CUA, the task hierarchy includes *scenarios* to describe the high-level context of the collaborative situation, *tasks* to indicate specific goals within the scenario, and low-level *task instantiations* to represent individual and collaborative ways of carrying out the task (see Figure 1).

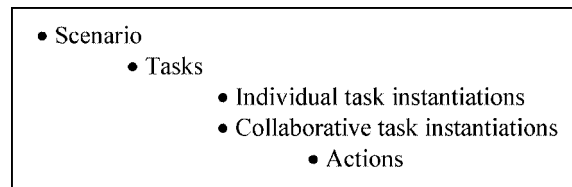


Fig. 1. Overview of the component hierarchy in the CUA task model.

CUA is oriented towards the particular needs of groupware evaluation, and as a result, it is less formal than other modeling schemes. Our intent is to provide groupware designers with enough contextual detail to help them to design and evaluate interfaces. This informal approach also helps us to control the added model complexity that is needed to represent the intricacies in group work. We fill any gaps in the contextual model by assuming that the design team has prior understanding of the work domain, and can make informed decisions based on observations, interviews, and common sense.

CUA models therefore provide a framework within which a knowledgeable evaluator can build a discount groupware evaluation. By providing a way for the evaluator to structure their explorations through the usage space, and by providing grounding for each collaborative interaction in a well-known operation, CUA is intended to help evaluators identify and resolve usability problems across the full range of use that the groupware interface will be expected to support. The main contributions of CUA are that it represents the breadth of uses that a groupware system may be expected to support, and that it grounds collaborative interactions in well-defined operations.

We now turn to the details of the CUA approach, starting with the idea of representing the mechanics of group tasks. In order to illustrate some of the details of CUA, we will refer to a simple illustrative example—a brainstorming meeting—as we discuss the larger issues. Brainstorming is a canonical activity for groupware systems, but is also an interesting activity to model because it is highly unstructured and there are a variety of repeated tasks (Figure 2 shows a scenario description for this activity).

4. REPRESENTING TEAMWORK AT A MECHANICAL LEVEL

The groupware evaluation techniques mentioned above define groupware usability as the extent to which a groupware system allows teamwork to occur—effectively, efficiently, and satisfactorily—for a particular group and a particular group activity. This means that our task models must be able to represent the elements of teamwork in a collaborative situation. However, the perspectives from which teamwork has traditionally been viewed in social science and CSCW literature do not provide us with a breakdown of the concept that can be used to assess specific interface designs used to carry out specific tasks.

We are in full agreement that these traditional perspectives (e.g. social or organizational) are crucially important for groupware design teams, in that they can help designers to understand the culture and context into which the system will eventually have to fit. As Grudin [1994; 1990] and others have

Scenario: Brainstorm ideas for an equipment list in a new computing laboratory

Activity description. A laboratory group convenes a meeting to think of a list of equipment that could be installed in their new lab in a new building. This is an initial meeting where no decisions have to be made, so the group is simply looking for ideas and a basic organization of those ideas into equipment categories. The group meets in their existing lab space, an open area with a large whiteboard at one end of the room. Two people act as primary scribes during the session, although others occasionally take the markers and write things themselves.

Roles. Participant, Scribe.

User specifications. The group is made up of the head of the lab, five graduate students, and two technicians. All members are computer scientists with experience in the vocabulary of computing equipment, and all (of course) are familiar with whiteboards.

Intended outcome. The overall purpose of the session is to generate a list of potential equipment for the new lab, roughly organized into categories. The lab head wants the list to be broad and to cover most of the new lab's requirements; individuals also have specific goals about making sure that particular pieces of equipment are mentioned.

Fig. 2. Scenario specification for a brainstorming task.

pointed out, many of the failures seen in groupware systems can be traced to a lack of understanding of organizational culture or of the social factors that affect the ways that people interact and work together.

Nevertheless, organizational-level or even group-level analyses of teamwork are poorly suited for some of the jobs that discount evaluation techniques are asked to do. First, the subtlety of social and organizational issues means that they must be investigated with actual users in the real work environment. However, discount evaluations are intended to be used early in the development cycle with low-fidelity prototypes that are far from functional and that cannot be realistically situated in the workplace. Second, discount evaluations are intended to provide design feedback on the specifics of the interface—feedback that can be used both to assess the current design and to help guide redesign for the next iteration. It is extremely difficult to tie social, organizational and political factors to the layout and operation of components in a groupware interface, thus we are interested in a way to represent the elements of teamwork at a lower level.

Our solution is to use a *mechanical* approach to analyze the interface. In particular, our conception of teamwork is derived from the affordances and constraints of the physical environment where the collaboration plays out: the shared workspace. We believe that there is a set of basic, core actions that happen in shared workspace collaboration, regardless of the organizational culture, the personalities of the group members, or even the type of task that is being carried out. These are mechanical actions like communicating with other members of the group, keeping track of what others are doing, negotiating access to shared tools or empty spaces in the workspace, and transferring objects and tools to others. Although these mechanics of collaboration are common and easily taken for granted in real-world shared workspaces, they are often difficult in groupware, and lack of support for these basic activities is often what makes groupware awkward and clumsy to use.

Using mechanical operations as our level of analysis allows us to target these specific kinds of usability problems in groupware. These problems will not, for the most part, interact with social and organizational concerns, such as organizational structure, group politics, or the personalities of group members. Even though social issues may still affect the system's eventual success or failure, our goal is that groupware development teams should at least be able to identify and solve basic mechanical usability problems before having to deal with more subtle organizational concerns.

Using a mechanical approach to analysis means that high level collaboration tasks, such as planning, arguing, and exploring are not seen in our framework. However, these collaborative activities can still be analyzed and specified with our approach by decomposing them into smaller, mechanical units. We will illustrate this with the brainstorming example that we will discuss throughout the rest of the paper.

While we stress the importance of mechanical analysis in CUA, we still include basic high-level information about the real-world work setting in our model. The mechanical approach to analysis does not imply that the high-level features of work are unimportant, but that by taking the analysis down to a basic level, we can consider work at both a high and low level of granularity. As we will discuss in section 5.1.1, we include scenario specifications as part of CUA, which include information about users, work settings, circumstances under which tasks are carried out, and the intended group outcome. This information does not capture the social and organizational complexities of the workplace, but it can be useful when considering the high level implications of design decisions.

4.1 The Mechanics of Collaboration

The mechanics of collaboration are the basic operations of teamwork—the small-scale actions and interactions that group members must carry out in order to get a task done in a collaborative fashion (see Table I). They are the things that will be common to a shared task even with a variety of social and organizational factors. We have presented the mechanics previously [Gutwin and Greenberg 2000]; however, they are an evolving set, and the version described here is a revised and expanded revised list. The mechanics are a useful level of analysis for evaluation-oriented task models because they provide a fine-grained view of teamwork; and since the mechanics are observable, collaboration can be analyzed and broken down into specific actions that evaluators can assess one at a time.

For task analysis purposes, the mechanics are our lowest level of representation for collaborative interactions. This implies that the mechanics are well understood by those who use the task model, since there will be no more detailed information about the activity within a mechanic. Thus, the mechanics are at the level both of common sense and common experience; since they are common to any number of shared workspace tasks, and since most people will have had a wide range of experience in shared workspace situations, evaluators should be able to use their existing knowledge (as well

Table I. The Mechanics of Collaboration

Category	Mechanic	Typical actions
Communication		
Explicit communication	Spoken messages	Conversational Verbal shadowing
	Written messages	Conversational Persistent
	Gestural messages	Indicating Drawing Demonstrating
	Deictic references	Pointing + conversation
	Manifesting actions	Stylized actions
Information gathering	Basic awareness	Observing who is in the workspace, what are they doing, and where are they working
	Feedthrough	Changes to objects Characteristic signs or sounds
	Consequential communication	Characteristic movement Body position and location Gaze direction
	Overhearing	Presence of talk Specific content
	Visual evidence	Normal actions
Coordination		
Shared access (to tools, objects, space, and time)	Obtain resource	Physically take objects or tools Occupy space
	Reserve resource	Move to closer proximity Notify others of intention
	Protect work	Monitor others' actions in area Notify others of protection
Transfer	Handoff object	Physically give/take object Verbally offer/accept object
	Deposit	Place object and notify

as their understanding of the work context) to simulate the operation of the mechanic.

The mechanics given in Table I are collected both from previous research into shared workspace collaboration [Clark 1996; Tang 1991; Hutchins 1990; Short et al. 1976; Bekker et al. 1995] and from our own experience with groupware systems [Gutwin and Greenberg 1996; 1999]. Together, these mechanics form a framework for mechanical action over a common workspace. They provide a well-defined way of conceptualizing and describing teamwork in groups, and this makes them well suited as an analytical component in CUA.

The mechanics cover two general types of activity: communication and coordination. Communication is broken into two categories: explicit communication and information gathering, and coordination is broken into two categories: shared access and transfer. In the following sections, we discuss the four categories and the mechanics within them.

4.1.1 *Category 1: Explicit Communication.* Explicit communication is communication that is intentional and planned; it is perhaps the most fundamental element of collaboration, and most of the other mechanics are based on it in some way. In shared workspaces, there are three main types of explicit communication: spoken, written, and gestural.

1A. *Spoken Communication.* Speech is the most common type of explicit communication in real-world shared workspaces. There are two main types that we wish to model. The first type is ordinary conversation where people engage in a dialogue about their work. The second type is *verbal shadowing*, the running commentary that people commonly produce alongside their actions, spoken to no one in particular but there for all to overhear. While conversation is used to communicate a specific message, shadowing helps other people stay aware of what that person is doing and why [Clark 1996]. Both of these types require that people be able to convey the message to a listener, but conversational speech also requires some indication that the message has been received and understood. In the brainstorming scenario, for example, a person might use ordinary conversation to state a new idea, either to another person, or to the whole group. A scribe at the whiteboard might also use verbal shadowing to give information about a larger activity—for example, “I’m just going to move these items over to make some more room...”

1B. *Written Communication.* There are also several types of written communications used in shared workspace collaboration. Conversations can be carried out through writing, and although this is rare in the real world, groupware systems often replace the voice channel with a text chat tool. A more common use of written communication is to send a persistent message, often in order to relay a message to someone who is not currently present. Here we are interested primarily in short-term persistence in the shared workspace rather than creation of reports or other long-term documents. In the brainstorming scenario, people use written communication to convey ideas and relationships on the whiteboard, and also to annotate or provide details for a particular item.

1C. *Gestural Communication.* In face-to-face work, gestures are frequent and are successful at communicating messages to others. There are several types of explicit gesture: pointing to indicate objects, areas, and directions [Tatar et al. 1991]; drawing to show paths, shapes, or abstract figures [Bekker et al 1995]; describing to show orientations, distances, or sizes [Bekker et al 1995]; and demonstrating to act out the use or operation of an artifact [Short et al. 1976; Tatar et al. 1991]. There are other more specialized types as well, such as the emblem, where a gesture stands for a particular word or phrase (e.g. thumbs-up for “OK”) [Short et al. 1976]. The critical requirements for gestural communication are that the sender has a medium that is rich enough to convey the gesture that she wishes to produce, and that the receiver is able to see the gesture with enough fidelity to interpret it. In the brainstorming scenario, people will use pointing extensively to indicate items on the whiteboard, and

may also use gestures to indicate paths between items, suggest groupings, and give directions.

1D. Combinations of Verbal and Gestural (Deictic Reference). When a verbal conversation involves objects in the workspace, the artifacts act as conversational props [Brinck and Gomez 1992] that let people mix verbal and visual communication. People use these props to simplify the task of referring to particular objects; this is deictic reference [Tatar et al. 1991], the practice of pointing or gesturing to indicate a noun (e.g. in the brainstorming example, a speaker might refer to a particular idea on the whiteboard simply as “that one”). Interpreting combined communication depends on knowledge about what objects are being discussed and what the sender is doing. In the brainstorming scenario, the most frequent verbal/gestural combination will involve pointing to give references to items on the whiteboard.

1E. Manifesting Actions. Actions in the workspace can also replace verbal communication entirely. However, manifesting actions must be carried out carefully to prevent them being mistaken as ordinary actions: the action must be stylized, exaggerated, or conspicuous enough that the “listener” will not mistake it [Clark 1996]. In the brainstorming scenario, one example of a manifesting action might be a person uncapping a marker in front of the whiteboard, an action that tells the others “I am about to write down another idea” without having to say so.

4.1.2 Category 2: Information Gathering. The second way in which information gets communicated in shared workspaces is through people gathering information from others in the space and from their activities. This is decoupled communication, since the producer of the information does not necessarily intend to communicate, and the movement of information is initiated by the receiver. Information gathering in shared workspaces is governed by two factors: first, the attentional focus of the gatherer (how hard are you looking), and second, the ‘volume’ at which the information is being produced by the producer (how loud are you talking or how large are your actions). We have identified four kinds of information that can be gathered.

2A. Basic Group Awareness. People maintain peripheral awareness of each other in shared workspaces in order to keep track of the basic organization of the collaborative session: who is in the workspace, where they are working, and (in general) what they are working on [McDaniel 1996]. This knowledge gives people a context for their own work, helps them make their communication more efficient, and helps them identify opportunities to assist another person or collaborate more closely with them [Gutwin and Greenberg 1996; Dourish and Bellotti 1992]. In the brainstorming scenario, people easily maintain basic awareness of who is in the meeting (by seeing the people in the room) and their general activities (e.g. by hearing verbal communication or by seeing actions like writing on the whiteboard).

2B. Activity Information from Objects (Feedthrough). More specific information about people's activities can be gathered by seeing or hearing the effects of manipulating objects in the workspace. This mechanic has been called feedthrough [Dix et al. 1998]. Examples include seeing changes to objects and inferring the activity that has gone on, and hearing the characteristic sounds of a tool (such as the squeak of a marker). In the brainstorming scenario, a person might see an alteration or annotation to an idea on a whiteboard (changes to an object) or might hear the squeak of a whiteboard marker and realize that something is being written (characteristic sounds).

2C. Activity Information from People's Bodies (Consequential Communication). Activity information can also be gathered by watching people's bodies in the workspace. This is consequential communication [Segal 1994], so called because the information is communicated as a consequence of activity. Many kinds of activity have characteristic and recognizable motions that can easily be seen and interpreted by another person. For example, in the brainstorming scenario the back-and-forth motion of erasing at the whiteboard can be understood even from a distance.

2D. Visual Evidence. When people converse, they require evidence that their utterances have been understood. In verbal communication, a common form of this evidence is back-channel feedback. In shared workspaces, however, visual actions can also provide evidence of understanding or misunderstanding [Clark 1996]. For example, as one person adjusts the way a picture is hanging in response to another person's directions, the adjustments themselves act as communication about whether the directions have been successfully received and understood. In the brainstorming scenario, visual evidence might be used during the process of indicating a location for a new item; as the scribe points to the whiteboard, the other person might say "further to the right... further..." using the scribe's location as evidence of how they are interpreting the directions.

2E. Overhearing Others' Explicit Communications. When others converse in the workspace, these conversations are available for all to hear even if the parties only intend to communicate with one another. Overhearing serves two purposes: first, it tells people that other members are present and are interacting; second, the content of the conversation may be valuable to the overhearer [Hutchins 1990]. In the brainstorming example, nearly all of the conversations are available to the group (and intentionally so) even if they are nominally between a subset of the people; the public discussions allow the group to be informed of opinions, agreements, and differences between others.

4.1.3 Category 3: Management of Shared Access. The third and fourth mechanics deal with coordination issues surrounding how objects within the workspace are accessed and used. Managing group access is a common coordination problem in a workspace where there are shared resources that are limited

in some way. These resources include work artifacts (e.g. a puzzle piece or a drawing), tools (e.g. whiteboard markers, scissors, or rulers), the workspace itself (e.g. an empty space on the board for adding a new item, or a corridor for reaching across a table), or even time (e.g. an opening in the 'airtime' of a conversation). We have identified three main activities where shared access issues occur.

3A. Obtain a Resource. Many shared resources have the notion of one-at-a-time use—objects that are too small to be manipulated by two people, most tools, and smaller work areas in the workspace—therefore people must act to obtain the resource for their own purposes. This means actually taking an object or tool into hand, or occupying a part of the shared space with their bodies. Since obtaining things in physical shared workspaces primarily involves reaching for them, this activity is coordinated in groups by people's ability to see which objects do not have people near them, and to see where others' hands and arms are moving. In cases where two people have both grabbed an object, the conflict is (usually) resolved by social protocols: people realize that they are in a resource conflict, and then one person releases the object. In the brainstorming scenario, for example, a group member must first obtain the marker from another person before recording an idea.

3B. Reserve a Resource for Future Use. In addition to obtaining resources for immediate use, people also often attempt to reserve objects and spaces for future purposes. For example, people will gather up several objects (e.g. different colors of marker) that they will make use of later on, or will move near a part of the workspace that they plan to work in next. The coordination of these reserving actions is similar to that for obtaining, but in general things are only brought into proximity rather than into hand. In the brainstorming scenario, a group member might reserve a part of the whiteboard that they wish to write on by standing next to it, blocking access with their body.

3C. Protect Your Work. When a person has completed work in a part of the workspace (e.g. created a set of artifacts or arranged them in some way) they often wish to make sure that others do not interfere with or destroy that completed work. This may be carried out through explicit communication (e.g. "don't touch that stuff") or by monitoring the area to see when anyone else moves into the area. In the brainstorming scenario, a group member can protect their work on the whiteboard from being changed or erased by placing their name next to the written idea to indicate ownership, by maintaining close physical proximity with a region of the board to indicate interest, or by monitoring other members to make sure they do not attempt to make changes.

4.1.4 Category 4: Transfer. The fourth mechanic concerns the movement of objects and tools between people. Being able to transfer things to another person is a crucial part of dividing the task, switching roles, and assisting others.

We have identified two ways that items are transferred in shared workspaces, differing only in the timescale of the interaction.

4A. Handoff. A handoff is a synchronous interaction where one person transfers an object or tool to another. The most common action is to physically give the object, and have the receiver physically take it. A second type is where the transaction takes place verbally—where responsibility or ownership of an item or a space is transferred, even if the thing itself is not. In the brainstorming example, a common type of handoff involves giving a whiteboard marker to another person so that they can write.

4B. Deposit. A deposit is an asynchronous type of transfer where one person leaves an object, file, or tool in a particular place for another person to retrieve later. This type of transfer happens primarily by moving the actual objects, although communication is usually required to identify the deposit location and to notify the recipient that the deposit has occurred. If the brainstorming activity is carried out asynchronously by physically distributed group members, each member might take a turn adding to a written idea list. To transfer the revised list, it might be necessary for the scribe to deposit the list in an agreed-upon location for other group members to retrieve later.

5. COLLABORATION USABILITY ANALYSIS

Collaboration Usability Analysis uses the mechanics of collaboration as the basic unit of a task model that can represent collaborative activities and collaborative variability. In this section we present the parts of the CUA task model in more detail and discuss the extensions that we have made to standard task-analysis techniques to represent variability in collaborative task execution.

5.1 The CUA Task Model

The major components of the task model are scenarios, tasks, individual and collaborative task instantiations, and actions.

- *Scenario.* High-level description of activities related to achieving a specific outcome. Scenarios contain: a high-level activity description, user specifications, a group goal, and a set of circumstances under which the scenario is carried out.
- *Tasks.* Basic components of scenarios, usually explicitly stated in scenario activity description. Describe what occurs in a scenario, but not how it occurs.
 - *Task instantiations (individual).* The taskwork component of a task.
 - *Task instantiations (collaborative).* The teamwork component of a task, specified as a mechanic of collaboration.
 - *Actions.* Common ways to carry out the mechanic specified in the collaborative task instantiation.

5.1.1 Scenarios. Task scenarios are commonly used as an evaluation tool in user testing and in discount usability engineering [Rubin 1994; Carroll 2000]. Scenarios are a descriptive formalization of the work that users perform in the real world—work that will likely be supported or that has bearing on the application that is being designed. A scenario typically contains multiple tasks and provides contextual information about the users and the circumstances under which the tasks are commonly carried out [Rubin 1994]. For our purposes, a scenario consists of the following elements: a high-level activity description, a user specification (user description and the knowledge users are likely to have), an intended outcome (the intended group goal in this case), and a set of circumstances under which the scenario is carried out [Nielsen and Mack 1994]. The user description in scenarios can specify the users in a generic way by describing the *role* they play in the group, or in cases where general descriptions are not warranted, the user description can describe the specific *actor* who is involved in the scenario. An example scenario description for the brainstorming task is shown in Figure 2.

The principal part of the scenario is the activity description, and this description provides details about the group's activities that are not necessarily captured in task diagrams. The description is a written narrative of the activities that take place in the scenario, and it can capture descriptive details about the tasks, users, and location. The specification also includes information about the goals and motivations of the group, the knowledge and expertise of the target users, and the real-world circumstances and constraints that shape how the group approaches the scenario.

In CUA, scenarios are built from data that is gathered from observations of the real-world work situation. Much has been written about how such observations can be conducted [Beyer and Holtzblatt 1998; Hughes et al. 1994], and it is likely that a large number of scenarios will be seen during the course of a typical field study. These scenarios can be organized if necessary into a diagram that indicates collaborators' workflow through the different situations (see Figure 6). In particular, scenarios for a shared task may or may not involve collaboration with other people; in CUA, however, it is the collaborative scenarios that we wish to investigate further. Once these have been identified, they are analyzed and specified in greater detail.

5.1.2 Tasks. Once the scenario is recorded, we can extract tasks from the scenario activity description and other observational data. Tasks are the main building block in the model; they are descriptions of individual work activities, and are often explicitly stated within a scenario (Figure 3 shows a task diagram for the brainstorming example). Task descriptions are oriented around outcome rather than process: they state *what* occurs in a scenario, but not *how* it is carried out.

5.1.3 Task Instantiations. To show how tasks are (or may be) carried out, a task is divided into *task instantiations*. In group work, tasks can often be carried out by different combinations of teamwork and taskwork. In CUA, we specify

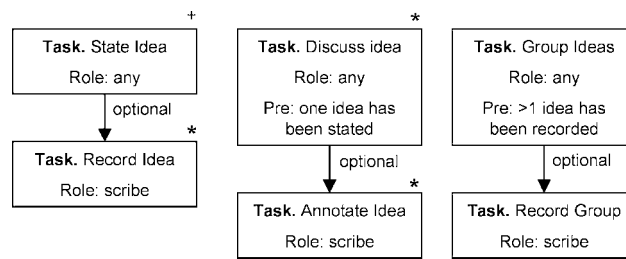


Fig. 3. Top-level task diagram for brainstorming scenario. Notation (see below) shows task sequencing, multiple start points, repetition indicators, preconditions, and optional tasks.

taskwork and teamwork separately as individual (taskwork) and collaborative (teamwork) task instantiations, but this is not to say that these processes are not tightly intertwined in group work. Taskwork, as previously discussed, consists of single-user activities that do not involve other group members, whereas teamwork involves the additional steps that must be carried out to complete a task in a shared manner. We model teamwork and taskwork as separate aspects of tasks because it is useful analytically to consider individual and shared work separately, particularly since levels of coupling between team members often vary during group work. In our experience, this separation is a useful approach since it fosters two analytical processes—consideration for specific individual actions each worker takes, and consideration for the mechanics of exchanges between workers. However, in acknowledgement of the interdependence that can be seen between teamwork and taskwork, individual and collaborative task instantiations can be modeled jointly as components of a single task (see Figure 8).

Individual task instantiations specify the steps in a task that group members carry out individually. For example, tasks such as signing a letter, searching for a name on a list, or highlighting a block of text do not necessarily require teamwork. In CUA, emphasis is not placed on modeling individual task instantiations in detail. However, since group members commonly switch between individual and shared work, individual task instantiations can be an important part of task analysis for groups. If finer granularity is needed for analyzing individual task instantiations, existing task analysis techniques can be utilized. For example, hierarchical task analysis techniques [Annett and Duncan 1967; Shepherd 1989] are well suited for this, since they can be applied to CUA individual task instantiations to analyze and specify at the desired level of granularity. It is also possible that individual task instantiations might depend on collaborative task instantiations. For example, an individual may need to monitor the actions of other group members while carrying out the individual instantiation.

Collaborative task instantiations represent the teamwork components of tasks, and they are specified using the mechanics of collaboration. Analysts use the mechanics to identify how collaborative activities take place. Using the mechanics can allow important but easily-overlooked aspects of a task, such

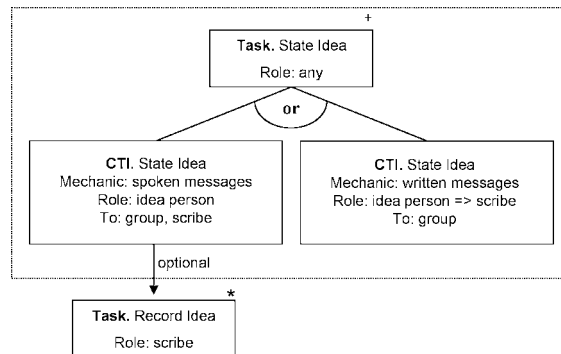


Fig. 4. Task flow diagram for 'state ideas' task showing alternate task instantiations.

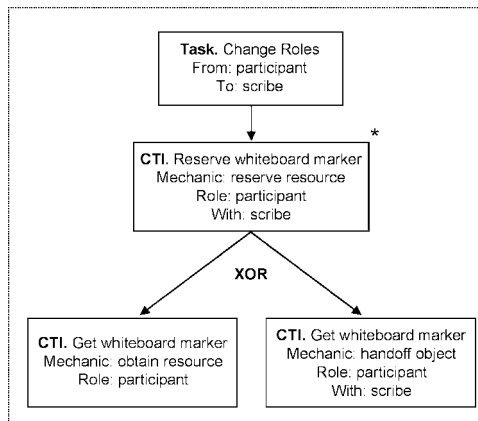


Fig. 5. Task flow diagram for 'change roles' task.

as consequential communication and visual evidence, to be identified so that support for the mechanics can be considered in the design. Figures 3, 4, and 5 show collaborative tasks and their associated mechanics.

5.1.4 Actions. The mechanics of collaboration do not provide a complete picture of task execution. Therefore, we include an additional specification in the task model. Each collaborative task instantiation can be carried out through a set of possible actions. Actions describe how the mechanic described in the collaborative task instantiation is carried out in the real world. For example, if a collaborative instantiation is to indicate an item on the whiteboard, common actions for accomplishing this may be drawing or pointing (both actions related to the gestural messages mechanic). Either of these actions is sufficient for accomplishing the collaborative instantiation; therefore, actions are usually presented as a list of reasonable alternatives (see Figures 8). Common actions for each mechanic are given in Table I.

5.1.5 Task Diagrams. In order for CUA to be a useful tool, the results of analyses must be represented in a way that is easily interpreted by those

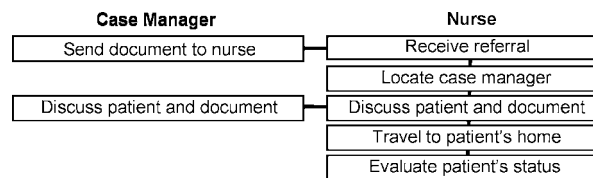


Fig. 6. Work flows and scenarios for a nurse and a case manager in a home care setting. Cells represent scenarios, and horizontal lines indicate collaborative scenarios that involve both individuals.

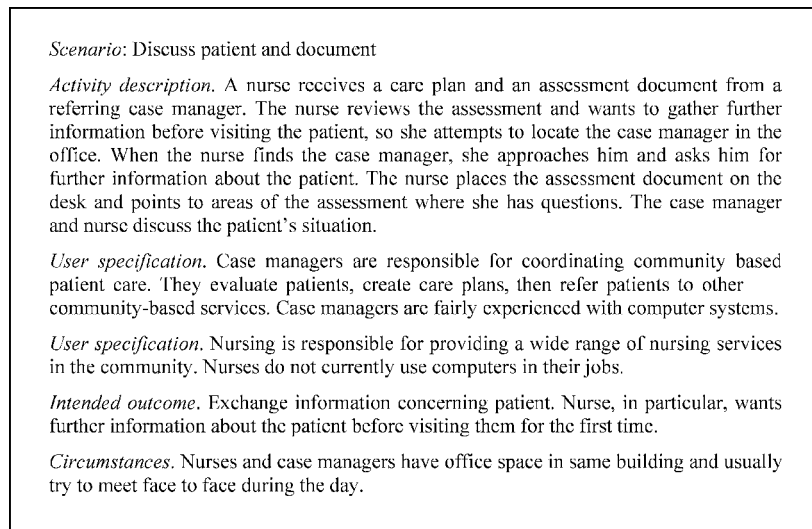


Fig. 7. Scenario specification for “discuss patient and document” scenario.

who will be involved in designing and evaluating a groupware application. In this section, we present an overview of diagramming techniques that we have found useful for organizing the results of analyses. Our approach to diagramming in CUA is not intended to provide comprehensive coverage of all possible task arrangements, but rather to provide a range of labels and descriptors that cover the majority of cases that are encountered when representing group tasks.

CUA diagrams can be built from the general to the specific, as group work is analyzed first at coarse granularity, and then at progressively finer granularities to bring out additional details. Typically, we begin by developing high-level diagrams of interesting sections of the group's workflow, and by showing how scenarios are typically sequenced within the group. At this point, scenarios are identified but are not analyzed or specified (e.g. Figure 6). Next, interesting scenarios can be specified (e.g. Figures 2, 7) and the tasks within the scenario can be diagrammed (e.g. Figure 3). Finally, interesting tasks can be diagrammed to bring out details about task instantiations and actions (e.g. Figures 4, 5, 8).

The structure of the diagrams captures three types of information: details about specific task components, a notion of the flow through task components,

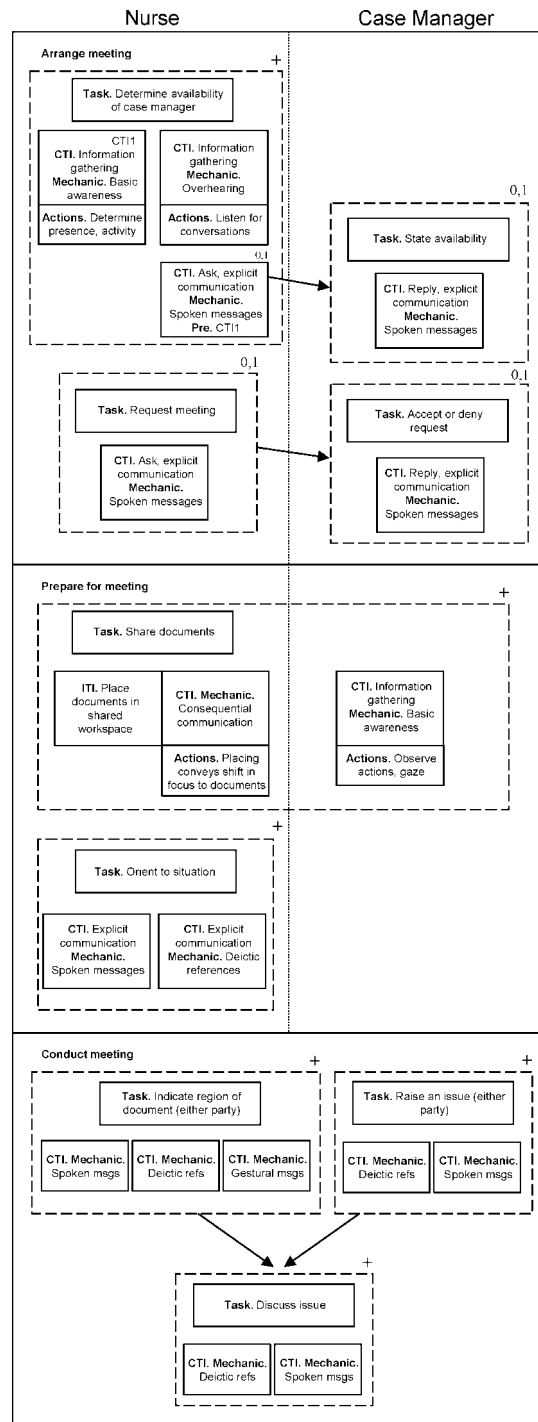


Fig. 8. Analysis results for "Discuss patient and document" scenario. CTI=collaborative task instantiation.

and a notion of how tasks are distributed among group members. Each task component is represented using a different cell in the diagrams, so a cell can correspond to a scenario, a task, a task instantiation, or an action, depending on the diagram's granularity. Since tasks, task instantiations, and actions are closely associated with each other, we find it useful to graphically group related components by placing them in a separate diagram or by placing a box around each task and its associated children (as shown in figures 4, 5, 8). In many instances, connectors are necessary in the diagram in order to represent the sequence of execution through the task components. We generally handle this by using arrows to indicate this flow when there is a strict sequence through the tasks; however, at times sequence is not important, and connectors can be omitted. As we will discuss in the next section, group tasks often have several variations in flow, including alternate tasks, optional tasks, and iterative tasks, and we will describe ways of representing these. Finally, in group tasks, it is necessary to represent which group members will carry out which tasks and task components. In some cases, this can be shown by dividing tasks in the diagram into columns, with one column allocated for each group member (see Figure 8). However, depending on the tasks and their distributions, this type of diagramming is not always practical, and the role or actor who will carry out the task can be specified within the task component's cell in the diagram (as shown in Figures 3, 4, 5).

In Table II, we provide an overview of the symbols and labels used in CUA task diagrams. We divide these into five categories depending on the type of information they provide: component type, flow, branching, iteration, and component allocation. Component type details common labels that are used to indicate the type of task model component that is represented by a cell in the diagram. Flow describes labels and symbols that indicate sequence of execution through the task components. Branching describes Boolean operators that are used to indicate which branches should be executed when multiple paths can be taken from a task component. Iteration provides regular expression operators that can be used to indicate how many times a task component cell can or must be repeated. Finally, component allocation specifies which group members are responsible for executing a given task component. In the next section, we will discuss flow, branching, and iteration in further detail.

5.2 Capturing Variability in Group Task Execution

A second main goal in the design of CUA is to capture the range of ways that a group task can be carried out. This is important, since the sequence and execution of group tasks can be highly variable. Group members work concurrently; they may interleave teamwork and taskwork, and variables such as division of labour and group composition may change, so that the execution of a given work scenario is likely to vary from time to time. During task analysis, capturing variability is important because it helps evaluators explore the range of ways that their system could be used for a particular activity. We are interested in broad coverage of the task space rather than minute detail because the goal of discount usability evaluation is to uncover major usability problems

Table II. Symbols and Labels Used in CUA Task Diagrams

Category	Symbol/Label	Description
Component type		
	TASK	Cell represents a task
	ITI	Cell represents individual task instantiation
	CTI	Cell represents collaborative task instantiation
	MECHANIC	The mechanic of collaboration associated with a collaborative task instantiation
	ACTION	Cell represents an action
Flow		
	↓ (arrow)	Indicates directional sequence of execution between components in the diagram
	PRE	Indicates a precondition that must be satisfied before a task component can be carried out
	OPTIONAL	Path connecting two components is optional
Branching		
	AND	All alternate branches must be executed
	OR	One or more branches must be executed
	XOR	Only one alternate branch can be executed
Iteration		
	*	Cell must be executed 0 or more times
	+	Cell must be executed 1 or more times
	(m, n)	Cell must be executed between m and n times
Component allocation		
	ROLE	A person who fills the specified role in the team must execute this cell
	ACTOR	A specified actor in the team must execute this cell

first. Representing the majority of uses allows evaluators to come into contact with a wide variety of potential usability problems. With these goals in mind, there are two major types of variability that we designed into CUA: variability in task sequence, and variability in the degree of coupling between group members.

5.2.1 Task Sequence Variability. Since by definition collaborative work involves several autonomous parties, group tasks can only rarely be specified as a concrete set of well-defined steps. Instead, there is considerable flexibility in how tasks get carried out, when tasks get executed during the session, and who performs each part of the task. In most group work scenarios, tasks may get left out, may happen in different orders, may be repeated a variable number of times, or may be performed by different people.

The hierarchical nature of the CUA task model accommodates many of the difficulties inherent in specifying task variability. First, it allows the representation of sequence and dependencies among tasks. Second, it allows for the representation of branching and variable paths through the task space at all levels of granularity (i.e. a scenario can be carried out with alternate tasks, a task can be carried out using alternate task instantiations, collaborative instantiations can be carried out with alternate actions). In practice, we expect people to develop paths for only major or important alternatives; while the task

model does allow the specification of fine-grained differences, this will usually be overly expensive in terms of the effort involved for the return.

Task Sequences and Dependencies. In some group work, activities must take place in a strict sequence since some tasks are dependent on the successful completion of others. Since tasks can be distributed among group members, this creates interdependencies between members and tasks alike. Dependencies can be conceptualized as pathways that must be followed to achieve the intended group outcome, and in cases where these dependencies are important, it is necessary to preserve the proper sequence of tasks and task components in the analysis results. CUA defines notation to indicate different types of sequencing.

For a simple task or a small group, it is possible to show sequences by linking task components with arrows (see Figures 3, 4). In other scenarios with larger groups or more complicated tasks, dependent task components may not follow immediately after each other, and the arrows can add substantial clutter to the task diagram. In these cases, it is more useful to model the *preconditions* for each task (and if necessary, each task component) in the task diagram. Preconditions can be stated in descriptive terms, or all tasks in a diagram can be numbered and then indexed as preconditions. For example, Figure 3 shows preconditions for two top-level tasks in the brainstorming scenario; in these tasks, the preconditions are simply written out.

Alternate Tasks. Some intended group outcomes might be successfully accomplished using more than one approach. When the analysis must accommodate alternate tasks and task components, the modeling of the task analysis results can be handled in several different ways. First, if the variation between different alternate approaches for accomplishing an intended outcome is only minimal, the alternate approaches can be analyzed and diagrammed in a single representation of the analysis results. To show that alternate paths can be taken through the task space, we provide representations for logical “OR” and “XOR” branching in our diagramming scheme. For example, Figure 4 shows how stating an idea in the brainstorming scenario could occur in two different ways with two task instantiations.

In other cases, the variability between alternate approaches for reaching an intended outcome may be substantial. In these cases, since it is difficult to capture the alternate tasks in a single diagram, multiple representations may be necessary.

Optional Tasks. Some tasks and task components may not be essential for the successful attainment of an intended group outcome, nevertheless, they may be regular or intermittent components in a group’s real world work activities. This is particularly true in the teamwork components of group work. Teamwork is often carried out to guarantee that each worker’s participation in the group is coordinated with the activities of others. However, at times, groups may choose to work closely together, or group members may work individually with minimal contact with others (this will be discussed further in Section 5.2.2). Therefore, some teamwork components may at times be optional (see Figures 3 and 4). For example, if a group is working together in a shared workspace,

teamwork may be necessary to coordinate and protect each group member's work. However, if each group member works separately but toward the same intended outcome, the same level of teamwork may not be necessary. In CUA task diagrams, optional task components can be indicated using the '*' symbol (which designates a task component as occurring zero or more times), or paths that indicate flow through the diagram can be labeled "OPTIONAL" to show that components that follow a particular path are not necessarily required. In Figure 3, for example, the scribe for the brainstorming group might or might not record an annotation on the whiteboard after the discussion of a particular idea.

Iteration. Group tasks and task components often occur in an iterative fashion. This does not differ substantially from single user tasks, but often the iteration that occurs in group work is necessary to carry out the teamwork aspects. Teamwork activities such as explicit communication and information gathering represent ongoing processes that are repeated over a period of time. For example, the tasks in brainstorming can occur any number of times, depending on how many ideas are generated and how much discussion is provoked about the ideas. By designating iteration for specific task components, some of the complexity required to specify group processes can be reduced. Notation for iteration follows that used for regular expressions: '*' indicates zero or more repetitions, '+' indicates one or more, and (m, n) indicates a specific range. Using this notation, Figure 3 shows that only the 'state ideas' task has to occur at least once; discussing and grouping ideas can happen any number of times but are not absolutely required.

5.2.2 Variability in Coupling. Coupling in collaboration is the degree to which people can work as individuals before needing to interact with another member of the group. In loosely coupled collaboration, group members can carry out more individual taskwork in between episodes of teamwork; in tightly coupled collaboration, interaction is more frequent and periods of individual work are shorter. In real world tasks, coupling is sometimes determined by the task (e.g. tasks that have strict role interactions or external time constraints), but can also vary depending on the group (e.g. some groups simply choose to work more closely than others do). Therefore, a task modeling scheme must be able to represent a variety of coupling styles, and must also be able to represent different couplings for the same task.

Taskwork components are typically the same, regardless of whether a task is carried out in a tightly or loosely coupled fashion. However, when a task is carried out in a tightly coupled fashion, those taskwork components may be distributed among several group members, and additional teamwork components must be added in order to capture details about how the group members collaborate during the task. This shift to shared, tightly coupled tasks may require the addition of collaborative task instantiations and actions in order to guarantee that the task can be carried out by multiple persons. Additionally, tightly coupled interactions may introduce interdependencies between workers that were not necessarily present during loosely coupled taskwork, and one

worker's actions may depend on the successful completion of another worker's actions. For example, brainstorming can be carried out by an individual group member or in a more tightly coupled fashion with other members of the group. The basic taskwork steps—record and annotate ideas—are the same regardless of the level of coupling. However, when the coupling increases, teamwork must be introduced in the form of explicit communication, transfer, and shared access (Figures 3, 4, 5).

When analyzing tasks and representing the analysis results in diagrams, it is up to the persons carrying out the analysis to determine a reasonable range of coupling variabilities that should be modeled. The observational data from the real world should provide some guidance in determining this. By representing a reasonable range of alternate teamwork components in task diagrams, a range of coupling styles can be considered in design and in usability evaluations. In addition to this, the task model provides a narrative component (a scenario specification as discussed in 5.1.1) that allows the specification of coupling details and information about variability that is not easily captured in a diagram. In the brainstorming scenario, the description might record two coupling styles from observations of the group: first, that the group sometimes prefers to discuss each idea as it is presented, and second, that they sometimes prefer to generate many ideas in parallel as individuals and then discuss each one afterwards.

6. A REAL-WORLD EXAMPLE: BUILDING AND USING A CUA MODEL

In this section we provide a real-world example of a CUA model, and a description of how it was used in a discount groupware evaluation. The example arises from design work that we have been carrying out with home care workers in a local health district. We are developing a groupware application that will allow home care teams to communicate, discuss documents, and hold case conferences.

We followed the analysis process described above to learn about the work domain and to specify tasks for home care workers. We interviewed members of each home care discipline, and then spent time with each discipline in the field, observing them as they worked. Once observational data were collected, we modeled the work scenarios for each home care clinician. In this example, we focus on two roles, a nurse and a case manager. Figure 6 shows an overview of the scenarios that were carried out by these two clinicians during one observation session. We inspected the high level workflows that were seen during our observations, and we discussed scenario composition, frequency and priority with home care workers in order to identify which scenarios were relevant to design work. Once we identified relevant scenarios, we applied CUA to analyze the scenarios at a more detailed level. The next sections show the results of our analysis for the “Discuss patient and document” scenario.

6.1 Task Model

In home care, mobile workers provide patients with healthcare services in their home. A patient can receive services from several different disciplines, including

nurses, case managers, and therapists. Workers spend most of their time out of the office, so they often do not see each other face to face. However, since multiple workers can work with the same patient, their actions are interdependent. In order for home care workers to move toward desirable outcomes, they must work together; one common collaborative activity involves examining a patient's file and discussing their needs.

6.1.1 Scenario. The scenario specified here (see Figure 7) includes two people: a case manager and a nurse. Prior to this scenario, the case manager generates a nursing referral for a new patient and places two referral documents in the nurse's mailbox: a care plan and an assessment document. Upon receiving these documents, the nurse wants to discuss the documents with the case manager.

6.1.2 Tasks. When we carried out a task analysis of this scenario, we found that there were three main divisions: the nurse had to arrange a meeting with the case manager, prepare for the meeting, then actually hold it. In Figure 8, we diagram critical tasks for the scenario by organizing them according to these three divisions. While there are other possible variations to this scenario that are not shown in the diagram, due to space limitations we limit our discussion to the variations that we observed in the field. At the top of the diagram, we show the task 'determine availability of case manager' which can take place in two different ways—through explicit communication or by gathering awareness information. Alternates of these types are common in many office settings where people can be easily observed. This task takes place in the larger shared workspace of the office itself. At the middle of Figure 8, we show a set of tasks in which the nurse introduces artifacts into a second workspace (a tabletop) and communicates basic information about what they wish to discuss with the case manager. Finally, we show a model of the discussion that makes up the actual meeting. This task uses a third shared workspace, the documents themselves, and involves the repeated tasks of indicating a part of the document, raising issues, and discussing them.

6.2 Using a CUA Task Model in a Discount Evaluation

We used the CUA models shown in Figure 8 for a discount evaluation of an early groupware prototype for the home care domain. The groupware application that we tested was one of a set of low-fidelity mockups that arose from early design work. We designed this and other prototypes in an attempt to support current work practices, and to facilitate collaboration and information sharing between workers when they are physically distributed. An illustration of the prototype is shown in Figure 9. In this section we summarize how we used the task models in carrying out a groupware walkthrough—a usability inspection technique for groupware—and where the task model added value to the evaluation process.

In the interface, documents relating to a particular home care patient are visualized on a timeline (at the top of the screen) that indicates the date they were generated and the discipline (e.g. manager, nursing, physical therapy)

of parallel tasks, and the common conceptual underpinnings provided by the mechanics of collaboration allowed us to consider how well we supported collaboration in tasks. We discovered a variety of usability problems, including:

- minor low-fidelity oversights such as the lack of an entry box in the text chat tool;
- other problems with explicit communication through the chat tool, such as the difficulty that a recipient might have in noticing that a message had arrived for them;
- that determining a person's availability through monitoring would be difficult because the system does not provide enough awareness information about presence;
- that a single shared workspace and a single text chat area do not allow multiple meetings of subgroups within the home care team; and
- that identifying items in shared documents through pointing can fail when the other person is scrolled to another part of the document.

The real-world context that was provided by the CUA task model helped improve the efficacy of our evaluation by allowing us to consider how well the prototype design supported workers in carrying out the work processes that are commonly used to attain the intended outcomes for the scenario. The exercise of stepping through the CUA model and considering how well tasks, task instantiations, actions, and flow were supported in the prototype allowed us to consider both taskwork and teamwork components of the scenario. We were able to consider the parallelisms of group tasks and variability in task execution, and we were able to identify a range of collaboration-specific usability problems.

By having task-based information about collaboration that is grounded in a mechanical level of analysis, we were able to identify specific problems that arise due to incomplete or mismatched support for the mechanics of collaboration. Stepping through the CUA results allowed us to identify a simple oversight in the prototype design. When we attempted to simulate collaborative task instantiations, we found that we did not include an entry box to allow users to add new messages to the text chat tool. We were also able to identify more complex collaboration problems. For example, by considering the “determine availability” task and its “information gathering” mechanic, we were able to ascertain that the interface needs to present users with more extensive awareness information. Additionally, the contextual information included in the scenario specification and in the workflow diagrams (e.g. Figure 6) provided us with a wider view of the work situation, and enabled us to identify a different class of usability problems. For example, workers may make use of desks and office spaces to secure privacy during conversations, so we identified the need for both public and private workspaces in our application.

7. DISCUSSION

Collaboration Usability Analysis has in our experience shown itself to be a valuable tool for groupware designers and evaluators. However, as with any technique there are limitations to the approach, and our work is shaped by our

original goal of supporting the needs of iterative groupware usability evaluation. In this section, we discuss issues that arise from meeting our objectives, and summarize the contributions of CUA that can be applied to group task analysis more generally.

7.1 Issues Related to the Mechanical Approach

We believe that the mechanics of collaboration are an innovative and useful way of characterizing group activity. To our knowledge there is no other analysis method that decomposes teamwork into elements that can be extracted from observations of real-world collaboration and mapped to components and structures in an interface. The mechanics are the critical enabling factor of CUA. Nevertheless, there are certain limits to the mechanical approach.

Understanding the Mechanics. We have presented the mechanics as activities that everyone has experienced in everyday group work, and for the most part they are common; however, certain of the mechanics are more esoteric and it is possible that the approach will not be easily used by all design teams. For example, groupware designers may not be familiar with communication types such as deixis, verbal shadowing, and manifesting actions. If these concepts are unfamiliar, then they will be difficult to extract from observations and difficult to use in evaluations. We have two answers to this problem. First, even these less well-known mechanics are still easy to understand, and we believe that most people will recognize these activities once they are explained (i.e. the unfamiliarity is with the terminology not the concept). For example, Baker et al. [2002] showed that people can learn to understand a variation of these mechanics in a modest amount of time. Second, even if designers do not cover all of the mechanics, it is the more frequent ones that are also more well-known (such as verbal communication, seeing changes happen, giving, and taking); this implies that the majority of activities can still be modeled, and that the most severe usability problems will still be found using this approach.

Difficulty Observing the Mechanics. In order for the task models to be built, it must be possible to determine which mechanics of collaboration are being used during observation of the real-world activity. For most of the mechanics (particularly explicit communication and transfer) this can easily be extracted from observations. However, some of the mechanics can sometimes be difficult to see. For example, information gathering can be done very subtly, in ways that are difficult to observe: it may be impossible to determine whether a person is noticing changes that are going on around them in the workspace. This means that other knowledge-elicitation techniques are important in consolidating knowledge about the mechanics and about how information flows through the collaborative situation. Techniques like interviews or contextual inquiry [Beyer and Holtzblatt 1998] can help the observers to validate their observations and add understanding of what mechanics are used for what collaborative activities.

7.2 Issues Related to Representing Variability

One of our goals for CUA was to be able to represent the range of ways that a group could carry out a task, so that evaluators could test more of the range of usage situations for their groupware interfaces. CUA represents two main types of variability—sequence variability and coupling variability—in its structures and notation. Although these additions do significantly increase the ‘carrying capacity’ of a task model, there are unavoidable tradeoffs that arise from the expansion in scope. The most important of these is that our models become less precise as they add possibility. This means that CUA cannot be used to build formal or executable models of group tasks, which means that the users of the models must exercise considerably more judgment and interpretation than with more restrictive models.

We believe that a less formal approach that is able to handle task variability is necessary for group work since the many factors affecting the execution of tasks make it extremely difficult to cover the territory while maintaining precision. The role of the evaluator who uses a CUA model, therefore, is to act as an expert arbitrator, able to prune the combinatoric expansion of possibility and choose reasonable paths for the evaluation. For example, the mechanics work differently in synchronous and asynchronous situations; so we depend on the knowledge of the evaluators to adapt their evaluation and role-play to the constraints of the overall situation. The responsibility given the users of CUA models implies that the value of the model is dependent on the person using it; but this is exactly the approach taken by all discount usability techniques. In walkthroughs, inspections, and heuristic evaluations, the principles and criteria used to judge an interface are always stated at a ‘motherhood’ level (e.g. “use simple and natural dialogue”), and the onus is on the evaluator to contextualize the principles for the task situation in question. “Know the user” is the cardinal rule in usability engineering, and it remains just as true for groupware development as for single-user software.

A second issue in representing variability is how the technique will scale to larger groups. Although the complexity of execution can potentially increase with the addition of more people to the group, we believe that complexity is controlled by the natural mechanisms of group management. In the real world, two things tend to happen as groups grow in size. First, the number of roles in the group tends not to increase (or not nearly as quickly) as the overall size, and CUA’s representation of roles rather than specific people allows models to include an arbitrary number of people who are playing the same role. Second, groups often simplify or linearize their activities as a self-limiting technique with larger groups: for example, in large meetings, people tend to adopt a one-at-a-time communication policy; this also serves to restrict the task models.

7.3 Generalizable Contributions of CUA

The main contributions of our work on Collaboration Usability Analysis are not the notation or the specific task structures, but the introduction of a set of concepts that are important for representing group work for the purposes of usability evaluation. The idea of looking at teamwork in terms of component

elements, the list of mechanics of collaboration, the idea of coupling variability, and the set of ways that task orders can be varied—we believe that these concepts will be valuable to groupware designers and others who have to model group tasks, regardless of whether they use the specifics of CUA or not. Previous conceptions of group work have concentrated at different levels of analysis, and have not considered these operational concerns; however, it is these that can have a significant effect on groupware usability. These concepts and principles can be detached from the CUA method, and could potentially be added to other existing techniques such as GTA [van der Veer et al. 1996] or could even be used to advantage in situations where very informal task analyses are carried out.

The second main contribution of CUA is the idea of a task framework that bounds the activities that can occur within a collaborative situation, and that provides both high-level and low-level contextual information to guide an evaluator as they inspect an interface. The task model provides an external record of the details of the users and the task, freeing the designer from having to remember everything about the work situation. Within the frame, evaluators can construct an evaluation plan that is appropriate to the needs of their current place in the development cycle. The descriptions and processes captured by CUA allow evaluators to be flexible, and to apply their knowledge of the work situation in an organized fashion.

8. CONCLUSION

In this article we have introduced a new type of task modeling scheme for collaborative tasks in shared workspaces. The scheme, called Collaborative Usability Analysis, focuses on the teamwork aspects of a shared activity, rather than the taskwork aspects. CUA is designed around the major requirements of discount usability evaluation in groupware development. First, CUA task models capture the breadth and variability inherent in a group task, helping evaluators remember and test the range of uses that their systems will see in a real situation. Second, CUA models use the mechanics of collaboration as their basic unit of collaborative interaction; this allows evaluators to consider how specific parts of the groupware interface support collaborative operations of communication and coordination. CUA allows groupware designers to use a wider range of discount usability evaluations, enabling the iterative design of groupware.

REFERENCES

- ANNETT, J. AND DUNCAN, K. D. 1967. Task analysis and training design. *Occup. Psych.* 12, 211–221.
- BAKER, K., GREENBERG, S., AND GUTWIN, C. 2002. Empirical development of a heuristic evaluation methodology for shared workspace groupware. In *Proceedings of the 2002 ACM Conference on Computer Supported Cooperative Work*. New Orleans, Nov., ACM Press, 96–105.
- BEKKER, M., OLSON, J., AND OLSON, G. 1995. Analysis of gestures in face-to-face design teams provides guidance for how to use groupware in design. In *Proceedings 1995 Symposium on Designing Interactive Systems*. Ann Arbor, Michigan, ACM Press, 157–166.
- BEYER, H. AND HOLTZBLATT, K. 1998. *Contextual Design: Defining Customer-Centered Systems*. Academic Press, San Diego CA.
- BIAS, R. 1991. Walkthroughs: Efficient collaborative testing. *IEEE Softw.* 8, 5, 94–95.

- BIAS, R. 1994. The pluralistic usability walkthrough: Coordinated empathies. In *Usability Inspection Methods*, Nielsen, J. and Mack, R. L., Eds. John Wiley & Sons, New York, 65–78.
- BRINCK, T. AND GOMEZ, L. M. 1992. A collaborative medium for the support of conversational props. In *Proceedings of the 1992 ACM Conference on Computer Supported Cooperative Work*. Toronto, Dec., ACM Press, 171–178.
- CUGINI, J., DAMIANOS, L., HIRSCHMAN, L., KOZIEROK, R., KURTZ, J., LASKOWSKI, S., AND SCHOLTZ, J. 1997. Methodology for Evaluation of Collaboration Systems. Tech. Rep. by The evaluation working group of the DARPA intelligent collaboration and visualization program, Rev. 3.0. <http://zing.ncsl.nist.gov/nist-icv/documents/method.html>.
- CARROLL, J. M. 2000. Introduction to the special issue on “Scenario-Based System Development.” *Interacting with Computers* 13, 1, 41–42.
- CLARK, H. 1996. *Using Language*. Cambridge University Press, Cambridge.
- DIAPER, D. 1989. *Task Analysis for Human-Computer Interaction*. Ellis Horwood, Chichester.
- DIX, A., FINLAY, J., ABOWD, G., AND RUSSELL, B. 1998. *Human-Computer Interaction*. Prentice Hall Europe, 408–412.
- DOURISH, P. AND BELLOTTI, V. 1992. Awareness and coordination in shared workspaces. In *Proceedings of the 1992 ACM Conference on Computer Supported Cooperative Work*. Toronto, Dec., ACM Press, 107–114.
- GREENBERG, S., FITZPATRICK, G., GUTWIN, C., AND KAPLAN, S. 1999. Adapting the locales framework for heuristic evaluation of groupware. In *Proceedings 1999 Conference of Computer Human Interaction Special Interest Group of the Ergonomics Society of Australia OZCHI'99*. Wagga Wagga, New South Wales, Australia, Nov. 1999. Available at <http://www.csu.edu.au/OZCHI99/>.
- GRUDIN, J. 1990. Groupware and cooperative work: Problems and prospects. In *The Art of Human Computer Interface Design*, Laurel, B., Ed. Addison-Wesley, 171–185.
- GRUDIN, J. 1994. Groupware and social dynamics: Eight challenges for developers. *Commun. ACM* 37, 1, 92–105.
- GUTWIN, C. AND GREENBERG, S. 2000. The mechanics of collaboration: Developing low cost usability evaluation methods for shared workspaces. In *Proceedings 9th IEEE International Workshops on Enabling Technologies: Infrastructure for Collaborative Enterprises (WET ICE'00)*. Gaithersburg, Maryland, Mar., IEEE Press, 98–103.
- GUTWIN, C. AND GREENBERG, S. 1999. The effects of workspace awareness support on the usability of real-time distributed groupware. *ACM Trans. Comput.-Hum. Inter. (TOCHI)* 6, 3, 243–281.
- GUTWIN, C. AND GREENBERG, S. 1996. Workspace awareness for groupware. In *Conference Companion on Human Factors in Computing Systems 1996 (CHI'96)*. Vancouver, April, ACM Press, 208–209.
- HOLTZBLATT, K. AND JONES, S. 1993. Contextual design: Principles and practices. In *Participatory Design: Principles and Practices*, Schuler, D. and Namioka, A., Eds. Lawrence Erlbaum Assoc., New York.
- HUGHES, J., KING, V., RODDEN, T., AND ANDERSEN, H. 1994. Moving out from the control room: ethnography in system design. In *Proceedings of the 1994 ACM Conference on Computer Supported Cooperative Work*. Chapel Hill, North Carolina, Oct., ACM Press, 429–439.
- HUTCHINS, E. 1990. The technology of team navigation. In *Intellectual Teamwork: Social and Technological Foundations of Cooperative Work*, Galegher, J., Kraut, R., and Egido, C., Eds. Lawrence Erlbaum, Hillsdale, NJ, 191–220.
- LEWIS, C., POISON, P., WHARTON, C., AND RIEMAN, J. 1990. Testing a walkthrough methodology for theory-based design of walk-up-and-use interfaces. In *Proceedings of the 1990 SIGCHI Conference on Human Factors in Computing Systems*. Seattle, Mar., ACM Press, 235–242.
- MCDANIEL, S. E. 1996. Providing awareness information to support transitions in remote computer mediated collaboration. In *Conference Companion on Human Factors in Computing Systems 1996 (CHI'96)*. Vancouver, April 1996, ACM Press, 57–58.
- NIELSEN, J. AND MACK, R. L. 1994. *Usability Inspection Methods*. John Wiley & Sons, New York.
- NIELSEN, J. AND MOLICH, R. 1990. Heuristic evaluation of user interfaces. In *Proceedings of the 1990 SIGCHI Conference on Human Factors in Computing Systems*. Seattle, Mar., ACM Press, 249–256.

- PATERNÒ, F., MORI, G., AND GALIBERTI, R. 2001. CTTE: an environment for analysis and development of task models of cooperative applications. In *CHI '01 Extended Abstracts on Human Factors in Computer Systems*. Seattle, Mar., ACM Press, 21–22.
- PATERNÒ, F., MANCINI, C., MENICONI, S. 1997. ConcurTaskTrees: A diagrammatic notation for specifying task models. In *Proceedings of Interact'97*. Sydney, Jul., Chapman & Hall, 362–369.
- PINELLE, D. AND GUTWIN, C. 2002. Groupware walkthrough: Adding context to groupware usability evaluation. In *Proceedings of the 2002 SIGCHI Conference on Human Factors in Computing Systems*. Minneapolis, Apr., ACM Press, 455–462.
- POLSON, P., LEWIS, C., RIEMAN, J., AND WHARTON, C. 1992. Cognitive walkthroughs: A method for theory-based evaluation of user interfaces. *Int. J. Man-Machine Studies* 36, 741–773.
- PREECE, J., ROGERS, Y., SHARP, H., BENYON, D., HOLLAND, S., AND CAREY, T. 1994. *Human-Computer Interaction*. Addison-Wesley Publishing, Reading, Mass.
- RICHARDSON, J., ORMEROD, T. C., AND SPEPHERD, A. 1998. The role of task analysis in capturing requirements for interface design. *Interacting with Computers* 9, 367–384.
- RUBIN, J. 1994. *Handbook of Usability Testing: How to Plan, Design, and Conduct Effective Tests*. John Wiley & Sons, New York.
- SEGAL, L. 1994. Effects of Checklist Interface on Non-Verbal Crew Communications, NASA Ames Research Center, Contractor Report 177639.
- SHEPHERD, A. 1989. Analysis and training in information technology tasks. In *Task Analysis for Human-Computer Interaction*, Diaper, D., Ed. Ellis Horwood, Chichester, 15–55.
- SHORT, J., WILLIAMS, E., AND CHRISTIE, B. 1976. Communication modes and task performance. In *Readings in Groupware and Computer Supported Cooperative Work: Assisting Human-Human Collaboration*, Baecker, R. M., Ed. Morgan-Kaufmann Publishers, Mountain View, CA, 169–176.
- STEVES, M., MORSE, E., GUTWIN, C., AND GREENBERG, S. 2001. A comparison of usage evaluation and inspection methods for assessing groupware usability. In *Proceedings of the 2001 International ACM SIGGROUP Conference on Supporting Group Work*. Boulder, Colorado, Sept., ACM Press, 125–134.
- TANG, J. 1991. Findings from observational studies of collaborative work. *Int. J. Man-Machine Studies* 34, 2, 143–160.
- TATAR, D., FOSTER, G., AND BOBROW, D. 1991. Design for conversation: Lessons from cognoter. *Int. J. Man-Machine Studies* 34, 2, 185–210.
- VAN DER VEER, G. C. AND VAN WELIE, M. 2000. Task based groupware design: Putting theory into practice. In *Proceedings of the 2000 Symposium on Designing Interactive Systems*. New York, ACM Press, 326–337.
- VAN DER VEER, G. C., VAN WELIE, M., AND THORBORG, D. 1997. Modeling complex processes in GTA. In *Proceedings of the Sixth European Conference on Cognitive Science Approaches to Process Control (CSAPC)*, Bagnara, I. S., Hollnagel, E., Mariani, M., and Norros, L., Eds. Baveno, Italy, Sept. Istituto di Psicologia, Roma, 87–91.
- WIXON, D., JONES, S., TSE, L., AND CASADAY, G. 1994. Inspections and design reviews: Framework, history, and reflection. In *Usability Inspection Methods*, Nielsen, J. and Mack, R., Eds. John Wiley & Sons, NY, 79–104.

Received June 2002; revised May 2003; accepted July 2003