ELSEVIER

Acta Psychologica 91 (1996) 297–322

acta psychologica

GTA: Groupware task analysis – Modeling complexity

Gerrit C. van der Veer [a,b,*], Bert F. Lenting [a], Bas A.J. Bergevoet [a]

[a] *CTIT, Twente University of Technology, P.O. Box 217, 7500 AE Enschede, The Netherlands*
[b] *Department of Computer Science, Vrije Universiteit, Amsterdam, The Netherlands*

Abstract

The task analysis methods discussed in this presentation stem from Human-Computer Interaction (HCI) and Ethnography (as applied for the design of Computer Supported Cooperative Work – CSCW), different disciplines that often are considered conflicting approaches when applied to the same design problems. Both approaches have their strength and weakness, and an integration of them does add value to the early stages of design of cooperation technology. In order to develop an integrated method for groupware task analysis (GTA) a conceptual framework is presented that allows a systematic perspective on complex work phenomena. The framework features a triple focus, considering (a) people, (b) work, and (c) the situation. Integrating various task-modeling approaches requires vehicles for making design information explicit, for which an object oriented formalism will be suggested. GTA consists of a method and framework that have been developed during practical design exercises. Examples from some of these cases will illustrate our approach.

PsycINFO classification: 4010

Keywords: User interface design; Task analysis; Task modeling; Ethnography; Groupware

1. Introduction

The main message of this paper is to show how task analysis methods from HCI can be combined with ethnographic methods as used in CSCW, for the purpose of constructing a complete task model that serves as a descriptive and analytic tool in designing complex cooperation technology. The methods from HCI and ethnography are

* Corresponding author. Tel.: +31 53 4893326.

SSDI 0001-6918(95)00065-8

first of all applied to collect information about the task domain, about problems, and about possible conflicting situations and activities. In order to model a complex task domain for the purpose of designing supporting technology, a framework of modeling concepts is needed that enables analysts to represent the most relevant results of the information collection for subsequent design decisions. A combination of concepts and techniques as advocated here is rather uncommon in current design methods, and, hence, needs to be based on an analysis of differences between approaches and on the needs of design as a generic activity aiming at serving end-users of information technology.

As soon as end-users are involved, the design of information systems cannot proceed without considering them. In fact, the discipline of HCI is now widely recognized as a valid partner in design methodology. The concept of 'task analysis' is often considered of central importance, although the exact meaning of the label varies widely depending on the design approach, like we will illustrate in Section 2.

For situations where different users have to collaborate through technology, or are in other ways effected by each other's work in relation to information systems, completely different design schools have been developed, indicated by concepts like CSCW or 'groupware'. In the latter types of design effort the label 'task analysis' seems to be less common, although the actual focus on the task environment in which the technology features is still a main part of the design effort as will be elaborated in Section 3.

The ways in which the task aspect of design is approached are based on theories and frameworks derived from different disciplines. The roots of the view on task knowledge often conflict, and the related methods do not seem to allow conversion to a common task model. Still, if we take the claims of the methods serious, we should not neglect the different standpoints. In Section 4 we will discuss a way to consider the differences and to derive from the different approaches an overall view on task knowledge.

Combining approaches from different disciplines requires a conceptual framework, derived from the domain of application, that enables the connection between concepts and data from various sources. GTA (Groupware Task Analysis), which offers such a framework, is defined in Section 5 and Section 6. The set of related concepts offered in Section 5 serves the purpose for complex task domains. In order to apply this analysis of the task structure, a method is needed to formalize the task knowledge for which in Section 6 an object-oriented representation will be proposed.

The actual way to combine different methods and formalisms has been developed in the course of 4 years of teaching user interface design. The design classes were organized in such a way as to simulate design in real industrial settings. Design teams of 10–25 designers were formed, existing of students from different disciplines (mainly software engineering, electrical engineering, psychology, artificial intelligence, business informatics).

The design teams were structured according to different design activities (Van der Veer et al., 1995), where groups of 2 to 5 students were responsible for a circumscribed part of design, like analytical task modeling (see Section 2), interaction analysis (Section 3), prototyping, usability testing, or management and negotiation with the client of the system to be designed. The current contribution focuses on task analysis, which is only part of the total structure of design activities performed in these courses.

The design examples chosen for the courses refer to complex situations (e.g., a

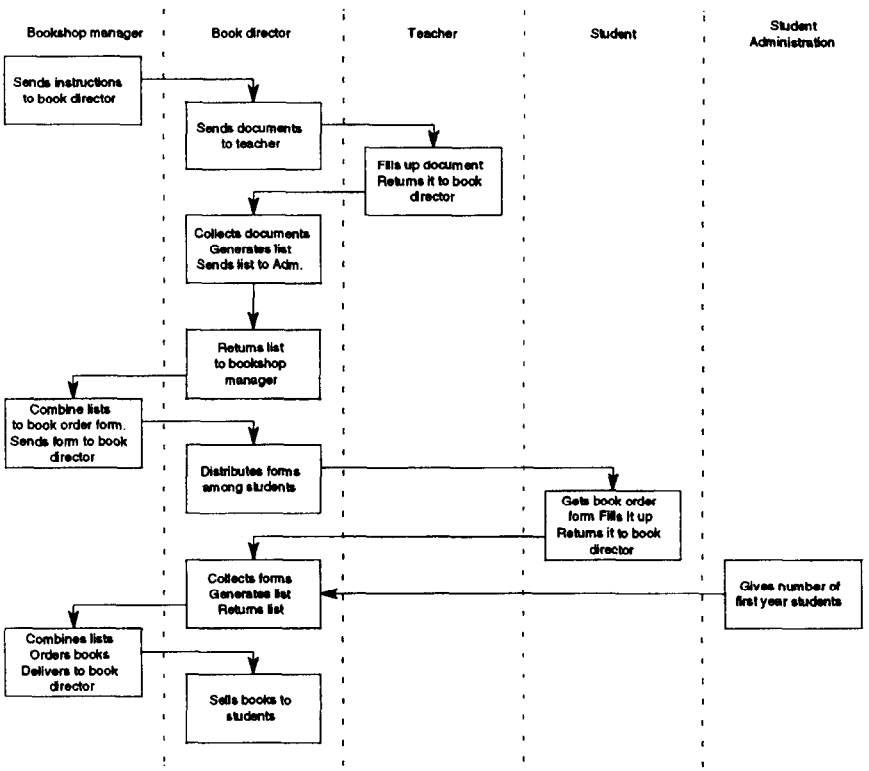


Fig. 1. Global work flow representation of the university book-selling case.

company electronic calendar and meeting organizer, a university book-selling system). In order to illustrate the complexity of this type of task domains, Fig. 1 presents a global view of the main work flow in the current situation of a university book-selling system (i.e., at the start of the design exercise). In fact, this work flow was only modeled after some ethnographic analysis had been performed (see Section 3). This analysis revealed that the book directors in each faculty were the key actors in the current process.

2. Task analysis in HCI design

Classical HCI features a variety of notions regarding task analysis. The concept of 'task analysis' is used to indicate different activities: (a) analyzing a 'current' task situation (the result of this activity will be referred to as task model 1), (b) specifying a task situation for which information technology is to be designed (task model 2), or (c) specifying the semantics of the information technology to be designed (the user's virtual machine). Many HCI task analysis methods combine more than one of these activities and relate them to actual design stages (Johnson et al., 1988). On the other hand, some

authors do not bother about the distinction: GOMS (Card et al., 1983) can be applied for any of them or a combination.

2.1. Analysis of the current task situation: Task model 1

In many cases the design of a new system is triggered by an existing task situation. Either the current way of performing tasks is not considered optimal, or the availability of new technology is expected to allow improvement over current methods. A systematic analysis of the current situation may help formulate design requirements, and at the same time may later on allow evaluation of the design. In all cases where a 'current' version of the task situation exists, it pays off to model this.

Sebillotte (1988) presents a method to collect task knowledge and structure this into a hierarchical model of subtasks. Scapin and Pierret-Golbreich (1989) elaborate on this method and provide an object oriented formalism for modeling knowledge of existing task situations, like Sebillotte mainly focusing on activities. The main method used by Sebillotte and Scapin, e.g., Méthode Analytique de Description (MAD) is a structured interview, focusing explicitly on the hierarchical relation of tasks. Fig. 2 shows part of the task structure of the university book-selling system analyzed by interviewing several book directors (one of the different classes of actors involved in this system).

In the fragment of the task in Fig. 2, the arrows indicate a decomposition of a task into subtasks. Labels printed along an arrow indicate a 'constructor' that specifies the conditional and temporal decomposition:

OR	the following subtasks are alternatives
COND	the following task is performed only if a condition is valid
LOOP	the following task is performed repeatedly
SET	the following tasks are performed in a non-determined order
PAR	the following tasks may be performed in parallel
SEQ	the following tasks are performed in a fixed order
	(the *default* constructor, not explicitly indicated in our figures)
SIM	the following tasks are performed simultaneously

Constructors can be combined (COND.LOOP indicates that under certain conditions the following loop is performed). Constructors of this kind represent the task structure as elicited by the interviews. In different task domains a slightly different set of constructors may be found.

Fig. 2 represents the knowledge of book directors about one of their tasks, i.e., the creation of a book inventory. This task is used in the current system to identify books that students can order for the next term (for which an order form will be created). To create the book inventory the actor waits until he receives a time schedule and order forms for the teachers (from the book shop). The form can be used by teachers to define one or more titles of books students should buy for the next term. The time schedule is used to synchronize activities between bookshop manager and book director. After all teachers are identified who will teach courses, the book director either transfers forms to

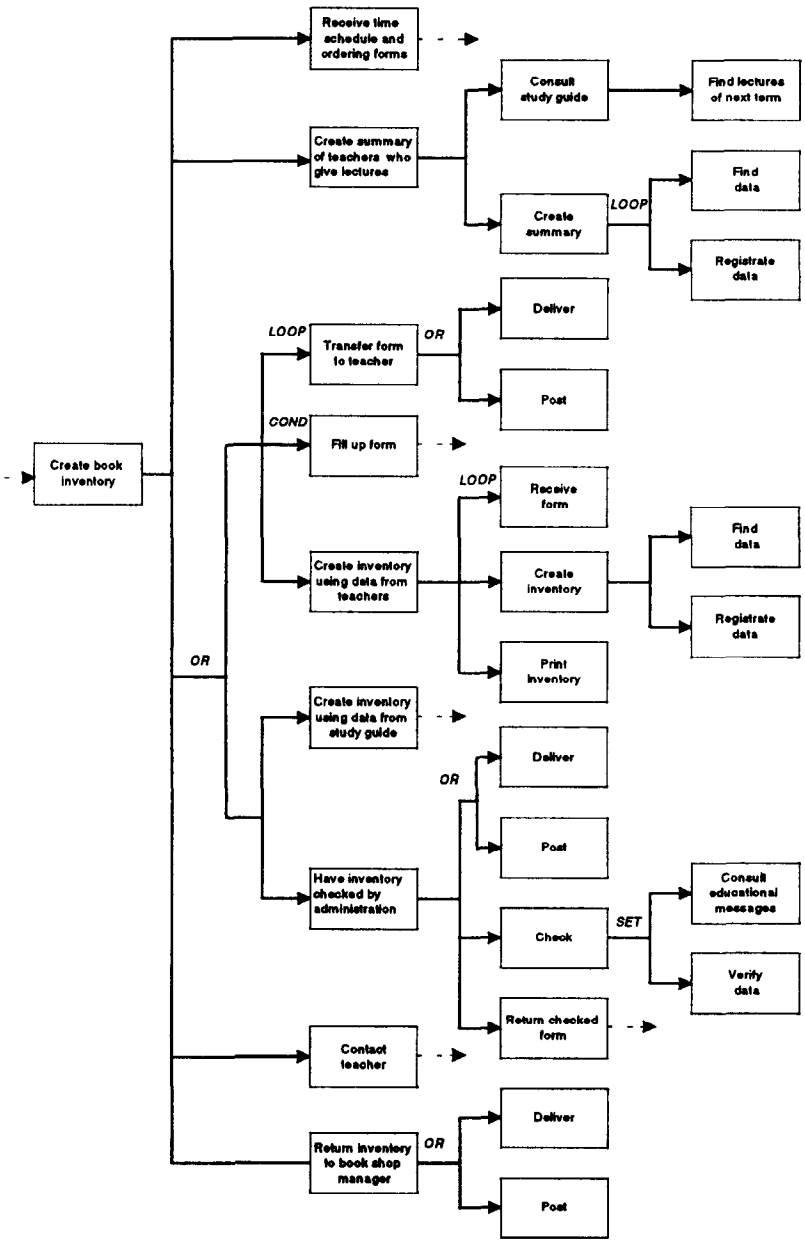


Fig. 2. MAD – fragment of the hierarchical task structure of book director tasks.

them, OR creates an inventory by himself (we will elaborate this complex subtask later on).

The complete inventory is finally returned to the bookshop manager, according to the

time schedule. MAD applies an object-oriented notation for the detailed description of each task (represented by a rectangle in Fig. 2). We will illustrate this notation in Section 4.

The information presented in the previous figure could be part of other HCI task analysis methods, e.g., Task Knowledge Structures (Johnson et al., 1988). Taxonomic substructures in TKS represent subtask structures as well as knowledge about objects (in the sense of 'things') and their semantic relationship. This knowledge can be seen as declarative. The objects are represented in an object-oriented formalism.

The various methods of task analysis in HCI and the related formalisms have much in common: a mostly hierarchical model of task structure, representing task knowledge of actors in the task domain. This model is in most cases completed by both procedural and declarative task knowledge related to, respectively, task decomposition and task-related knowledge of the semantics of objects.

Task models of this type pretend to describe the situation as it can be found in real life, by asking or observing people who know the situation from experience (e.g., Johnson, 1989). Task model 1 is often considered of a generic nature (e.g., Sebillotte), indicating the belief of authors in this field that different expert users have at their disposal basically the same task knowledge.

2.2. Specification of the future task situation: Task model 2

Many design methods in HCI that start with task modeling are structured in a number of phases. After describing a current situation (task model 1) the method requires a re-design of the task structure in order to include technological solutions for problems and technological answers to requirements. Johnson et al. (1988) provide an example of a systematic approach where a second task model is explicitly defined in the course of design decisions. The design decisions that lead from task model 1 to task model 2 will in actual situations be based on three different sources, which each provide specifications for a new situation.

2.2.1. Problem specifications

The actual system, as modeled in task model 1, will often be considered (by the actors, i.e., the spokesmen that provide the input for the first task model) far from ideal. Their knowledge and attitudes, as far as made explicit during the first phase of task modeling, will help the designer to specify parts of the task structure, and characteristics of task-related objects, that require a change to optimize the task performance. In our design practice we discovered that classical HCI methods for collecting task knowledge will only reveal part of the problems, though. Ethnography (see Section 3) may provide complementary specifications. We will elaborate this in the course of this paper.

2.2.2. Client's specifications

The requirements as stated by the 'client' (the instance that 'pays' the design team for improving the task situation) have to be clearly distinguished from the actors' problem specifications. Clients (although they will sometimes also be actors in the task domain and users of the technology to be considered) may provide completely different

aspects for design, like economic considerations, time constraints, and usability specifications for acceptance testing. On the other hand, clients may show concerns related to goals that seem to be irrelevant to the primary tasks of the system concerned. In the book-selling case, the client (the chairman of the university student association) stated:

> "By selling books to our members we try to establish a better contact with them. At least they visit our offices and get an idea of the services we offer them ... We are not just book sellers, we are an association where members should participate."

In as far as clients' specifications are in conflict with proposed solutions to problem specifications, the designers will have to negotiate with the client.

2.2.3. Technical specifications

Technology plays an important role in the definition of specifications. On one hand, it serves as a base for the designers' insight needed to meet specifications from the previous sources, i.e., what technological possibilities can serve a solution. On the other hand, technology itself defines requirements (constraints), i.e., what solutions are feasible. Design decisions related to the application of new technology, the introduction of new artifacts and the definition of new work procedures will consider technical feasibility as well as the specifications derived from knowledge of the actors and from requirements of the client.

Providing this structure of sources of design specifications does not automatically lead to task model 2, and does not guarantee optimal design decisions. It merely serves as a base for structuring problems, alternative solutions, and criteria, and, hence, can be helpful in developing an explicit design rational.

Task model 2 will in general be formulated and structured in the same way as the previous model, but in this case it is not considered a descriptive model of users' knowledge, although in some cases it might be applied as a prescriptive model for the knowledge an expert user of the new technology should possess.

2.3. Specification of supporting technology: The user's virtual machine

The third type of modeling activity that may be found in HCI design concerns the technology to be designed. In principle, this might be considered part of task model 2 (e.g., Card et al., 1983, in the case of GOMS). However, in other HCI approaches the actual design activities focus on the technology as such (e.g., Tauber, 1990). In this part of design the activity produces a detailed description of the system as far as it is of direct relevance to the end-user.

Oberquelle (1984) introduces the concept 'virtual machine' to indicate "the functionality of the system ... where implementation details and details of the underlying hardware are suppressed". Tauber (1988) elaborates the concept of the user's virtual machine (UVM) which indicates the total of user-relevant knowledge on the technology, both semantics (what the system offers the user for task delegation) and syntax (how task delegation to the system has to be expressed by the user).

We will borrow the term UVM to separate the design of technology (as far as

```
Conceptual events:

type [EVENT > SEND CANCEL_MESSAGE]

    description:  for [TIMESLOT: *t]
                      [event.CREATE ([CANCEL_MESSAGE],[TIMESLOT])
                      [event.SEND ([CANCEL_MESSAGE],[PARTICIPATION_LIST])
    precondition:     [state.FILLED([TIMESLOT: *t])]
    comment:        "create cancellation message for timeslot t and send it to all participants."

end [SEND CANCEL_MESSAGE].

type [EVENT > CANCEL MEETING]

    description:  for [TIMESLOT: *t]
                      [event.SEND CANCEL_MESSAGE([TIMESLOT])]
                      [event.COPY MEETING TO TODO_LIST([MEETING],[TODO_LIST])]
                      [event.EMPTY TIMESLOT([TIMESLOT])]
    precondition:     [state.FILLED([TIMESLOT: *t])]
    comment:        "cancel meeting in timeslot t"

end [CANCEL MEETING].
```

Fig. 3. ETAG – fragment specification of an electronic calendar and meeting organizer.

relevant to the end-user) from the design of the 'new' task situation as a whole, mainly because the UVM models the detailed solution in terms of technology, where task model 2 focuses on the task structure and work organization. In actual, design iteration will be needed between the specification of these two models, which should be an explicit activity, making the implications of each obvious in its consequences for the other.

For the specification of supporting technology HCI has delivered several different formalisms. ETAG (extended task-action grammar, Tauber, 1988) is a modeling approach that allows design decisions to be specified in most relevant details (though not all, e.g., the presentation components are not modeled in detail), and it relates most smoothly to the type of models HCI applies for the previous design phases. An ETAG specification of an information system can be conceptualized as the knowledge of the system that a fully competent user should have in order to operate the system in delegating tasks.

Conceptual events (Fig. 3), as part of an ETAG description, are presented with their procedural description and the preconditions for enabling the execution of the event. As can be seen, a conceptual event can be defined as a structure of other events. This resembles the definition of tasks by a structure of subtasks, as mentioned for task models 1 and 2. In the case of the UVM, however, the events are defined as to be executed by the technology, and in ETAG they are specified as far as relevant for the future user of this technology, without bothering with implementation. E.g., in Fig. 3 is shown that the cancellation of a meeting will consist of sending a cancellation message to the participants, of making a copy of the meeting note in a 'to do list' and of clearing the slot in the calendar. In this example this is all the user needs to know about this event – the implementation of details should not concern him.

2.4. HCI task models are knowledge models

All HCI task modeling is rather narrowly focused, considering mainly individual people's tasks, although Johnson (1989) considers the aspect of roles and the phenomenon of allocating subtasks to different actors. Most approaches are based on cognitive psychology (e.g., Johnson refers to knowledge structures in long-term memory, Tauber refers to 'knowledge of competent users') and all refer to knowledge as can be modeled after individuals who are knowledgeable or expert in the task domain, whether this domain already exists (task model 1) or still has to be re-structured by introducing new technology (task model 2 and the UVM).

3. Design approaches for CSCW

CSCW work stresses the importance of group phenomena and organizational structure and procedures. Most CSCW approaches are at least partly rooted in Activity Theory or related theories (Nardi, 1996), as much as most HCI approaches are related to Cognitive Psychology. Activity Theory stresses the importance of the interrelations between actors, objectives, and the community in the effects of using tools, applying rules, and the division of work. Hence, in analyzing complex activities (a concept that can be considered equivalent to the concept of task or goal in HCI task analysis), the 'classical' HCI methods are considered insufficient, focusing only on activities and knowledge of individual actors. CSCW literature strongly proposes ethnographic methods.

Ethnography literally means describing culture. When ethnographers are engaged in the design of complex technology ('groupware' or 'cooperation technology') they study a task domain (or 'community of practice') by becoming a participant observer, if possible with the status of an apprentice, being accepted as an outsider in this respect and being themselves aware of their status of analyzing observer. The ethnographer observes the world 'through the spectacles of the aboriginal' and at the same time is aware of his status of an outside observer whose final goal is to understand and describe for a certain purpose and a certain audience (in the case of CSCW: a design project).

Ethnographers apply different methods. In our experience with applying ethnography in design of technology for complex task domains one method turned out to be both reliable (i.e., leading to reproduceable results in the description and interpretation of task phenomena) and relatively easy to teach to designers of technology: 'interaction analysis' (Jordan, 1994). In this method the analysts, like in all ethnographic approaches, start their observation purposely without a conceptual framework regarding characteristics of task knowledge. Systematic data collection in interaction analysis starts with video recording of relevant phenomena (the relevance of which can only be inferred from prior observation) followed by systematic transaction analysis, where inter-observer agreement serves to improve reliability of interpretation.

The analyst (ethnographer) has to choose a focus (or, if needed, more than one) on either activities, environments, people, or objects. The choice of focus is itself based on prior ethnographic observations, which illustrates the bootstrapping character of this type

of analysis. Knowledge of individual workers in the task domain may be collected as far as it seems to be relevant, but it is in no case a priori considered the main source, and will never be considered indicative for generic task knowledge.

Fig. 4 shows a fragment of an interpreted person-oriented interaction analysis record, analyzed as part of the case of the university book-selling system. The record shows a general description of the situation, including a graphical representation (Fig. 4a) of the spatial layout of the environment and the different actors (indicated by circles) and objects, followed by a time stamped transcription (Fig. 4b) of the video tape as interpreted after consent between different interpreters (some of which were not present at the recording – as suggested by Jordan).

In this actual example it was found out that the same actor, when interviewed according to the MAD method, revealed only the top levels of the task hierarchy and could not be persuaded to externalize more detailed knowledge. The ethnographic record showed many details of task performance, as well as revealed the relation of his activities to both the activities of other actors in the task domain and to situational conditions that were never discovered with HCI methods. An example of this concerns the reception and collection of ordering forms from the teachers.

The MAD model (Section 2) represents the actors' verbal statements that this normally occurs once, so that the creation and printing of the inventory has to be done also only once. From the ethnographic analysis could be found that the process of collecting forms and updating the inventory in certain circumstances (as when the fall term brings many last minute changes in programs) occurs more than once. Printing, in this case only, is held back until all ordering forms have been processed and checked against the actual situation of course planning (which may involve contacting study coordinators and teachers).

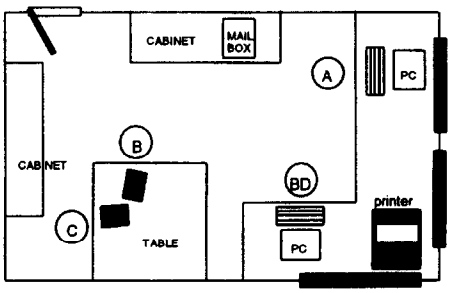
The ethnographic approach is unique in its attention to all relevant phenomena in the task domain that are not explicitly verbalizable by (all) experts, including knowledge and intentions that are specific for some actors only, conflicting goals, cultural aspects that are not perceived by the actors in the culture, temporal changes in beliefs, situational factors that are triggers or conditions for strategies, and non-physical objects like messages, stories, signatures and symbols, of which the actors may not be aware of their functions in interaction.

When analyzing the different records from book directors in other faculties, it turned out that book directors act in two different ways (as was indicated in the MAD task hierarchy represented in Fig. 2 by the OR constructor). Ethnography shows that some book directors will transfer the empty forms to the teachers and process them on return, as originally planned for the total organization. In a single faculty, however, the book directors decided to fill the ordering forms themselves on the base of the course catalogue for this faculty, which has in this situation turned out to be very reliable. Moreover, in this situation the forms will subsequently be checked by the student

Fig. 4. (a) Ethnography (interaction analysis) – person-oriented record (environment). (b) Ethnography (interaction analysis) – person-oriented record (transcription).

(a)

The data recording is a person oriented record, where the BOOK DIRECTOR (BD) is the person being followed. The recording starts when the BOOK DIRECTOR enters his office which is outlined in the figure below.



Person A sits behind a computer and is working with it. Persons B and C are negotiating. The black rectangles on the right of the figure indicate three windows. The door to the office is in the upper left corner.

(b)

Time (minutes)	Actions (*italics* denote spoken language)
0:00	The BOOK DIRECTOR enters the office. The BOOK DIRECTOR walks to B and C and says: *Hi girls!* B says: *Hello Archie*, while C says: *Hi!* The BOOK DIRECTOR then walks to A and says: *Hi Richard, did you manage to have those invitations printed yet?* A replies: *I'll go to the printer this afternoon; I'm making some final adjustments now.* The BOOK DIRECTOR says: *That's fine.*
1:30	The BOOK DIRECTOR walks to his mailbox, placed on top of a cabinet, and gets his mail. He collects all ordering forms from teachers and puts back the other mail.
4:18	The BOOK DIRECTOR takes the ordering forms and takes place behind a computer (see figure). The computer is already operational and shows a menu. He opens a disk box containing about 40 disks, and searches through the floppy disks. He takes a disk labelled 'Book inventory' out of the box and puts it into the computer's disk drive.
4:47	The BOOK DIRECTOR starts up a word processor application from the menu and opens a file from the floppy disk. This is done by selecting a menu option 'Open' and selecting the file to be opened. An inventory of all book titles that have to be ordered appears on the screen.
5:14	The BOOK DIRECTOR takes one ordering form and looks at it. He positions the cursor on the right place in the inventory by moving the mouse and starts entering the data from the ordering form.
5:29	The BOOK DIRECTOR enters the code of the lecture, the name of the lecture, the teacher's name, the book title, the author, the publisher and the year in which the book has been published. All this information is read from the ordering forms filled in by teachers.
6:54	After this the BOOK DIRECTOR takes the next ordering form and processes this form in the same way.
10:38	After three ordering forms the BOOK DIRECTOR writes the inventory file back to the floppy disk. He does so by selecting the 'Save' option from a menu. He pulls the floppy out of the disk drive and puts it back into the disk box on the same position he took it earlier. He closes the word processor application. The menu reappears on the screen.
11:08	End.

administration of the faculty concerned, which in this particular case is both willing to do so and well equipped for this service.

Ethnography often helps discovering conflicting goals by regarding the history of a community of practice that leads to the current situation. Book directors need accurate data about books needed for the next term before a certain date (set in the time schedule from the book shop). Teachers feel the need to inform the book director when they are going to change their course.

If all remains identical to the past, teachers often do not feel the need to return the form. When an 'ethnographer' in our design team found this, he explicitly asked a teacher, who explained:

> "I get such a form very infrequently. A year has three terms, in some terms I don't even lecture. If I have nothing to order or I won't even be using a book, I throw the order form away. If I do need a book, I contact the bookshop manager, to find out if it is available and I order it then at once."

The teacher in this case has a different perception of his task from the book director, and he neglects the role of the latter completely in approaching the book shop directly. The bookshop manager, when in turn asked about this, was found to be very motivated to talk about this 'problem' as he perceives it:

> "The past has learned that teachers sometimes neglect forms and order books directly by calling our office. They are not aware of any deadlines or delivery dates, which causes me a lot of trouble with both teachers and publishers. Furthermore they disturb the complex administration that is done by the book directors. Unfortunately there is no organizational rule that could prohibit teachers from such awful behaviour. I have in the past complained a lot about this because I have to serve two instances instead of one. I get really tired of teachers who order out of date. It causes me financial risks about which I have to negotiate with the University. This does not improve my relationship with the University."

The book director, in the case of a no return, has to make sure he understands that either there is no need for any book for this course, or the entry from last year is still valid.

Ethnography will reveal relevant 'cultural' aspects that influence task structures. Personal contact between students and the students association turned out to be a strong motivating factor for the book directors to keep up with the current system, even if they could imagine much more efficient centralized book distribution systems. Some situational factors show to be very individualized. In analyzing a person-oriented record it was found that a book director on a single occasion went in person to the teacher to get a book ordering form filled. The reason for this behaviour became obvious from a recorded meeting in the teacher's room: he (i.e., the book director) wanted to discuss some personal business, which in fact absorbed his attention so much that he had to be reminded by the teacher to complete his original task.

Ethnography asks for interpretation from the analyst to a much larger extent than HCI data collection methods do. This will often allow ethnographic methods to detect non-physical objects, like was found in the university book-selling case: one of the book

directors applied his personal lecture schedule (which he knew by heart and did not even maintain anywhere on paper) as the main source for creating a teachers summary from which he could start contacting the teachers. All others consulted relevant documents like study guides. In the (MAD type) interview with this actor, the need for a source for a list of relevant teachers was never indicated.

Interaction analysis covers the methods for information collection and subsequent interpretation that might serve as a basis for developing task model 1 (and no more than this since this specific method only covers information on the 'current' state of a task domain). However, the methodology for the structuring of the collected data and interpretations into a total task domain description is often rather special and difficult to follow in detail. The general impression is that CSCW design methods skip the explicit construction of task models 1 and 2 and, after collecting sufficient information on the community of practice (and its history), immediately embark on specifying the UVM, based on deep knowledge of the current task situation that is not formalized. This might cause two types of problems: on the one hand, the relation between specifications for design and analysis of the current task world might depend more on intuition than on systematic design decisions; on the other hand, skipping task model 2 may lead to conservatism in view of organizational and structural aspects of the work for which a system is to be (re)designed.

4. Design of groupware requires complex task modeling

In situations of computer support for cooperative work design methods should be used by technology designers (hardware and software, including distributed processing, networking, multimedia and hypermedia). The result of design will be used in complex task domains and has to be understood by end-users in complex organizational structures. Hence, a systematic design approach should start with an analysis and description of the current task world and the specification of the future users' task world, including the community of practice.

4.1. A need for explicit models

For the benefit of the design, all design decisions should be based on 'external' modeling (models that can be communicated and are represented for that purpose) in at least a semi formal sense. Task model 1 should provide a representation of the current task world in such a way that it both covers aspects of the task world that are relevant for the users and for the community of practice, and that it can be managed in the design process. Task model 1 should relate to the subsequent representations of task model 2, the task world to be designed including (possibly new) task decomposition procedures, task allocations to people and technology, communication structures, management procedures. Task model 2 should relate to the specification of the system to be designed in the sense of a (more or less) formal representation of the UVM. In the current paper we will not elaborate details for modeling the UVM.

The reason for a certain level of formality for these three models is the fact that the

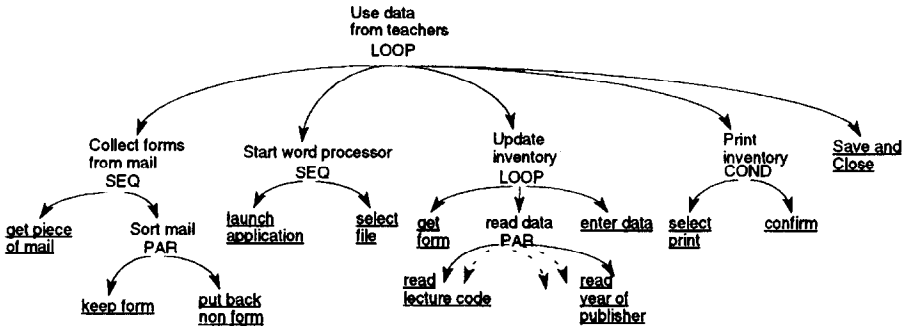


Fig. 5. MAD based – representation of interaction analysis fragment from Fig. 4b.

designing engineers need to communicate between the different stages (between themselves and in design teams), be able to back-track design decisions, and have representations available that allow them to keep design decisions explicit. If a consistent choice of formalism is made, this could, moreover, help design by smoothing the path between different phases and providing notations that could be supported by design tools. At the current state of the art of software engineering, 'object-oriented' notations might be an obvious choice.

In fact we discovered in our design examples that a transcription of the ethnographic interaction analysis into a formal model is possible. Fig. 5 presents a MAD-based task structure based on the book director record from Fig. 4.

In order to be more precise, transcription of ethnographic records of procedures into TKS or MAD type formalisms turns out to be feasible. Fig. 6 presents a MAD type 'task object' for the subtask 'update inventory' for the same record.

This representation illustrates an attempt to represent ethnographic data in an object oriented formalism. It allows the formal representation of detailed information like the type of objects (textlines) used by the respective actors and a description of the relevant change from initial state to final state in terms of these objects.

Interestingly, HCI methods like MAD even offer a completely object-oriented method for representing alternative ways of decomposing a task. Fig. 7 shows a task-type hierarchy along the formalism of Scapin and Pierret-Golbreich (1989).

Task-object-N°1	Update inventory
Kind-of	Update-task
Initial State	Textlines 1,2,3.....,M
Final State	Textlines 1,2,3..... ,M+k
Goal	Add 1 Order
Condition	Access ? (file) = true
Structure	*LOOP.SEQ* arguments (get, read, enter)

Fig. 6. MAD based – task object after the subtask 'update inventory' from Figs. 4 and 5.

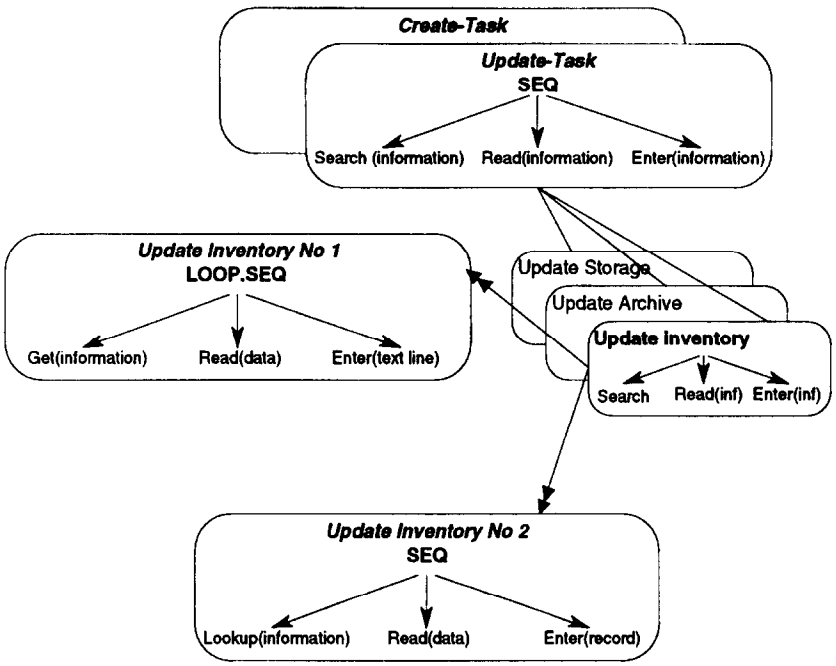


Fig. 7. MAD based – representation of semantic relation between book director task objects.

In Fig. 7 single headed arrows indicate a 'part of' relation. Double headed arrows indicate instances of the same class (i.e., alternative ways of decomposing the same generic task), and simple lines indicate subtypes.

These representations, on the one hand, help dealing with task knowledge in an interdisciplinary design team, but, on the other hand, fail to reveal the situatedness of the different alternative task decompositions. Apparently a formal way of representing task knowledge is needed but the formalism has to be reconsidered in relation to the complexity of the task world concerned with groupware applications.

4.2. Knowledge in complex task worlds

For the benefit of the users, the task models used in design should cover aspects of the task world and of the technology to be designed that are relevant for the knowledge and learning of the individual users, for the group processes and management in the task world, and for the task situation. The methods of investigating actual task situations and users should be fit for the aspects to be modeled, as well as chosen in relation to the nature of the phenomena that are relevant in the task situation.

In complex situations of people at work, possibly using artifacts, not all phenomena that are relevant for task description are of the same type. For instance not all might be available as verbalizable knowledge of individuals, and not all may be reliably perceived by a trained observer or documented. Hirschheim and Klein (1989) elaborate two

task world knowledge	**individual**	**group**
explicit	**a.** knowledge and skills	**c.** models/stories/instructions
implicit	**b.** intuition/expertise	**d.** culture/community of practice

Fig. 8. Dimensions of knowledge of complex task domains.

dimensions of task knowledge (world knowledge in their terminology): subjectivist–objectivist and order–conflict.

The order–conflict dimension stresses the fact that the analyst is not a passive outside observer, but active in shaping the task model. If different actors in the task world hold different beliefs or conflicting knowledge and goals this should, first of all, be explicitly reflected in the task model 1, and, secondly, be considered in specifying task model 2 where conflict might be either resolved by forcing the new task situation into a unified solution of the conflict, or, alternatively, by providing multiple possibilities for procedures and private situations as part of the general task world for individuals with conflicting interests.

The subjectivist–objectivist dimension of Hirschheim and Klein indicates the problem of collection of knowledge for task model 1. In this respect, we refer to a framework by Jordan (1994), see Fig. 8.

Relevant task domain information may have to be collected focusing on different phenomena, using different methods of data collection. Based on an analysis of the character of the knowledge sources in this framework, different methods are identified to collect all information needed to construct task model 1.

For task knowledge in cell a of Fig. 8, psychological methods will be used including those elaborated by Johnson (1989) and Sebillotte (1988): interviews, questionnaires, think-aloud protocols, and (single person oriented) observations. For knowledge indicated in cell b observations of task behaviour will have to be complemented by hermeneutic methods to interpret mental representations (Van der Veer, 1990). For the knowledge referred to in cell c the obvious methods concern the study of artifacts like documents and archives. In fact all these methods are to be found in classical HCI task analysis approaches.

The knowledge indicated in cell d is unique in that it requires ethnographic methods like interaction analysis. Moreover, this knowledge can be in conflict with what can be learned from the other sources, as is already shown in the examples presented in the previous Sections. First of all, explicit individual knowledge often turns out to be abstract in respect to observable behaviour, and turns out to ignore the situatedness of task behaviour. Secondly, explicit group 'knowledge' (e.g., expressed in official rules and time schedules) often is in conflict with actual group behaviour, and for good reasons. In fact, official procedures do not always work in practice and the literal application of them is sometimes used as a political weapon in labor conflicts as a legal alternative for strike. In all cases of discrepancy between sources of task knowledge, ethnographic methods will reveal unique and relevant additional information that has to be explicitly represented in task model 1.

The allocation of methods to knowledge sources should not be taken too strictly. In fact the knowledge sources often cannot be located completely in single cells of the conceptual map. The main conclusion is that we need these different methods in a complementary sense, as far as we need information from the different knowledge sources. The kind of information we need is the topic of the next section.

5. Conceptual framework for GTA

The framework for groupware task analysis that is presented here is based on comparing the different approaches mentioned earlier, and on an analysis of existing and proposed systems for HCI and CSCW. The conceptual framework in its current state should be considered tentative. The structure needs to be validated and possibly augmented by studying examples of design exercises and products of design for HCI and CSCW. The framework might also need to be extended if relevant analytic methods require other distinctions than can be based on this structure and its elaboration in formal modeling methods.

The framework as such is intended to structure task models 1 and 2, and in elaborating the different aspects we indicate the methods that are needed in order to collect the information in the case of task model 1. Obviously, for task model 2 design decisions have to be made, based on problems and conflicts that are represented in model 1, in combination with requirement specifications as formulated in interaction with the client of the design. For a discussion of these design activities, see Van der Veer et al. (1995).

Task models for complex situations need to be composed of different aspects. Each describes the task world from a different viewpoint, and each relates to the others. Consequently, the resulting final task model will be redundant at the level of representation for human readers. This will allow designers to read and to design from different angles, and provide slots for design tools to guard consistency and completeness. The three viewpoints (focus on people, work, and situation, respectively) that we will apply in our approach are a logical derivation from the main focal points in the domain of HCI as well as CSCW. Both design fields consider people ('users' vs. 'cooperating users' or user groups) and work (activities or tasks, respectively the objectives or the goals of 'interaction' and the cooperative work). Moreover, especially CSCW stresses the situation in which technological support has to be incorporated. In HCI the situation (in the restricted sense of environment, see Section 5.3.3) is sometimes only implicitly considered. In this section we will elaborate our conceptual framework and at the same time indicate how the relevant aspects of task knowledge may be acquired.

5.1. People

The first aspect focuses on people (human actors), both individual and in groups. People are considered in relation to the task world, hence, we need to make a distinction between individuals and the roles they play. Moreover, we need the concept of organization of people.

5.1.1. Actor

This label refers to individual persons. Important for task modeling is to identify relevant types of actors, and to characterize them on relevant characteristics. Types may be identified based on two different types of variables: Psychological characteristics like cognitive styles or spatial ability (Van der Veer, 1990) and task related characteristics like expertise or knowledge of information technology. For the detection of which types are relevant, ethnographic methods may be needed, for the general characteristics of actors in a certain task world and for the description of relevant types of actors psychological methods will be the first choice.

5.1.2. Role

Roles indicate classes of actors to whom certain subsets of tasks are allocated, by free choice or as the result of the organization. By definition roles are generic for the task world. More than one actor may perform the same role, and a single actor may have several roles at the same time. Roles may be performed temporarily, be negotiated between actors and accepted or refused. Actors may have internal (mental) representations of their own roles and others' roles and roles may be represented externally by instrumental or symbolic behaviour and by objects (white coat, stethoscope, wig). Both for the detection of roles and for the description of role-related task aspects and external representations, ethnographic methods are best fit, although TKS (Johnson, 1989) provides at least the appropriate notion. For the acquisition of knowledge on actors' internal representations of roles psychological and hermeneutic methods have to be applied.

5.1.3. Organization

'Organization' refers to the relation between actors and roles in respect to task allocation. The organization describes the human structure in the community of practice. Part of the organization is generic (as far as the structure of roles is concerned), another part concerns the current episode in the history of the task world (the organization as far as dependent on current individual actors and the roles they currently perform). The identification and subsequent description of the organization may be done by archival and document study, complemented by TKS and ethnographic methods. This last method will often be needed since actual organizations turn out to evolve, and divert from official or prescribed states, partly due to the effect of technology in use.

5.2. Work

Some approaches refer to goals as the unit of description of work (GOMS: Card et al., 1983), but we prefer to focus on the structural as well as dynamic aspect of work, hence, we will take 'task' as the basic concept. This is no big difference, since tasks and goals in most frameworks have a one-to-one relation. In activity theory tasks are referred to as 'actions' (which are, like in HCI task analysis approaches, considered to be hierarchically structured), where long-term tasks are referred to as 'object' or 'motive' (Nardi, 1996). We will make a distinction between tasks and actions in the 'classical'

HCI terminology, and, moreover, we will elaborate task structure and the structure-related concepts of protocol and strategy.

5.2.1. Task

Tasks can be identified at various levels of complexity. The unit level of tasks needs special attention. Payne and Green (1989) call this the 'simple task', but this notion may either indicate an artifact of a system, or a psychological concept, which sometimes results in ambiguity in analysis. We need to make a distinction between the lowest task level that people want to consider in referring to their work, the 'unit task' (Card et al., 1983), and the unit level of task delegation that is defined by the tool that is used in performing work, like a single command in command-driven computer applications. This last type of task we will call 'Basic task' (Tauber, 1990). Unit tasks will often be role-related.

Complex tasks may be split up between actors or roles. Unit tasks and basic tasks may be decomposed further into (user) actions and (system) events, but these cannot really be understood without a frame of reference created by the corresponding task, i.e., actions derive their meaning from the task. Activity theory and 'classical' HCI are in complete agreement in this respect. Tasks and task structures may be detected by analytic methods (Sebillotte, 1988) as far as they fit into the category of explicit individual knowledge or individual behaviour. Otherwise observation and ethnography may be needed. The description needs to be procedural, decomposing them either into a structure of subtasks or into a structure of actions.

5.2.2. Task structure

The task structure will often at least partially be hierarchical. For the indication of temporal order and dependency structure, concepts like the 'constructors' of Scapin and Pierret-Golbreich (1989) are relevant (see Section 2.1). Task structures for task model 1 are not always known by single actors, mainly when different roles are involved in performing different subtasks. On the other hand, performance on certain subtasks may influence the procedures for other subtasks. In this case, the identification of the task structure will require a combination of HCI methods and ethnography.

5.2.3. Actions

Actions are identifiable components of basic tasks or unit tasks, which have a meaning in performing a unit of work, but which derive their meaning only from the task they are part of. For instance hitting a return key has a different meaning depending on whether it concludes a command, or confirms the specification of a numerical input value. The speech act of confirmation has a different meaning depending on whether it follows another person's question or command. On the other hand, actions are the smallest elements of a basic or unit task that change or define the meaning of that task. In describing actions, the goal is to identify the meaning, not the physical characteristics.

In Activity theory these components seem to be equivalent to 'operations', which are at the level of automatism and the elements of subconscious feed-back loops. This theory stresses the phenomenon that actions may become operations by continued

learning and experience and that they must become 'actions' again when the operations are frustrated.

Typical actions in HCI and CSCW are the specification of objects or events, and speech acts. Actions may aim at changing (or operating on) attributes, 'location' or existence of objects, change attributes of the environment, or may effect mutual task performance between different actors. Actions that concern the 'content' of an object may often be considered to act on other objects that are contained in the current object ('themes', see below).

Actions, as parts of basic tasks or unit tasks, are often not explicitly 'known' (i.e., verbalizable) or actors are reluctant or unable to be very precise in this respect. Observation and ethnographic methods will be the main source of information.

5.2.4. Protocols

This concept indicates actual 'rules' as turn out (via observation and ethnography) to be applied for decomposing tasks, to be distinguished from 'rules' that may be stated explicitly in instructions which are sometimes not actually followed. Protocols may be 'situated', i.e., the environment and the presence of actors with certain roles may constitute conditions for protocols to be triggered. In the book-selling design example we found an example of a 'rule' (part of the official printed instruction for book directors):

> "The order form included with this letter should be copied and forwarded to the teachers who give lectures at your faculty. After the order forms have been returned they should be combined into a total order form (the inventory) that presents all book titles needed. This total order form should be delivered to me (i.e., the bookshop manager) personally before August 1, 1994."

In the analysis of this design case we found one book director who in fact completely complied to this rule, for which he gave the reason: "I am doing this job for the first time, so, please, be patient with me." In our terminology this is a protocol since it is actually applied, even if only by a novice in the role concerned.

5.2.5. Strategies

'Strategies' indicate structures that can be considered protocols used mainly by experts or typically preferred by them. These structures will often be situated in the same way as protocols are. Strategies may have started from explicit problem-solving and knowledge-formation episodes and subsequently have become implicit expert knowledge. Hence, observations and ethnographic methods will be needed to analyze them. Strategies will be role related. In the book-selling case we found an example of a strategy with an experienced book director (performing this role for the third year already) who adopted the use of a database application for the subtask 'update inventory' (instance No 2 in Fig. 7). When asked for the reason he used a database system (as was found during ethnographic observation) he stated:

> "When I took over the job from the previous book director he told me a lot about the problems with the forms. As a student in Computer Science I'm interested in

information systems. The step to use a database application comes from the fact that I use the same system at home."

This protocol illustrates the situatedness of strategies.

5.3. Situation

Analyzing a task world from the viewpoint of the situation means detecting and describing the environment (physical, conceptual, and social) and the objects in the environment. Object description includes an analysis of the object structure.

5.3.1. Object

Each thing that is relevant to the work in a certain situation is an object in the sense of task analysis. In this framework, 'objects' are not defined in the sense of 'object oriented' methods. Objects may be physical things, or conceptual (non-material) things like messages, gestures, passwords, stories, signatures. Non-material objects, and sometimes physical objects as well, may in the task situation be referred to by external representations of different character: verbal labels, graphics, metaphors, gestures. Actors that perform a certain role may be objects in a task situation and will be labeled 'active objects'. Non-human system components like computer-based agents may also be active objects.

The identification of relevant objects will depend on the condition of knowledge (explicit or implicit) and on whether the object figures in a task for a single person or in group situations. Relevant objects may be used to transport meaning and information between different agents without any of them being aware of the objects' nature (e.g., anecdotes that contain strategic information). As far as explicit knowledge is involved, analysis of verbal material from archival sources or from interviews may be of help, starting with the identification of nouns in relation to task references. For implicit knowledge about objects, observations and ethnographic methods have to be used, both for detection and for description.

5.3.2. Object structure

In order to describe the semantics of objects, two kinds of relations between object types have to be identified.

1. Object types are related via a type hierarchy, indicating sub-type–super-type relations. Sub-types inherit the characteristics of their super-type as far as no further specifications have to be added. Analysis will reveal the exact relations of object types of certain levels in a type hierarchy featuring in the task world.
2. Semantic relations between object types may metaphorically be indicated by place relations, where a certain type of object can be 'in' or 'on' another object type (Tauber, 1990, uses the concept 'theme' for this relation in his ETAG formalism), and where objects may 'move' from one place to another (each place being provided by an object). Apart from the relation between object types, objects will be related to tasks as agent (active objects), as subject, or as featuring in conditions of task structures. The identification of object structures will be an analytic (HCI type)

activity, based on verbal protocols from actors and on systematic observation of the situational relations in which objects are used.

5.3.3. Environment

The task environment is the current situation for the performance of a certain task. It includes actors with roles, conditions for task performance and for strategies and protocols, relevant objects, and artifacts like information technology that are available for subtask delegation. The history and temporal structure of relevant events in the task situation is part of the actual environment. The environment features as condition for task structures (inclusive protocols and strategies as far as these are situated). The analysis and description of environments often will need ethnographic methods.

6. Formalism for GTA

Collecting knowledge along the three viewpoints mentioned is only part of the game. In order to construct a task model (first of all task model 1) formal methods have to be used to combine the information in a clear and unambiguous knowledge structure that is available for further activities in design. Based on design considerations mentioned earlier, we developed GTA as an approach that complies with state of the art software engineering methods, i.e., using an object-oriented notation. We do not prescribe the exact nature of the formalism, but only indicate the types of objects (in this context we do NOT refer to 'things' in real life but to objects in the sense of an object-oriented notation).

The general feature of our formalism is that we define a generic object type for some

Object: timeslot	
superordinate: memo	**subordinate:** day_slot hour_slot quarter_hour_slot
themes: meeting holiday business travel	
places: month calendar week calendar day calendar	
relevant tasks: cancel meeting initiate meeting postpone meeting receive meeting cancellation forward meeting cancellation	
actors/roles; competence meeting participant; initiate cancellation meeting initiator; prohibit cancellation	
passive/active: passive	
attributes: time, date	

Fig. 9. Object template – example from the electronic calendar and meeting organizer.

of the concepts elaborated in Section 5, e.g., *task* (including both simple tasks and structured tasks like protocols and strategies), and *object* (including objects that feature in certain situations as 'environment').

The description of these object types refers to other concepts of our framework, creating a closed structure of knowledge (with a lot of redundancy, needed for reasons of readability by human designers). Figs. 9 and 10 present object-oriented templates for objects (in the meaning of 'things') and tasks.

Fig. 9 shows the template for objects, where the slots have been filled based on the design exercise for a calendar and meeting organizer. The slots for superordinate and subordinate refer to the place of each object in the object type hierarchy ('time slot' is a subtype of the more general type 'memo', day_slot, hour_slot, and quarter_hour_slot are subtypes that inherit all characteristics of the current object definition unless explicitly changed in the definition of these objects).

The themes and places slots indicate the semantic structure (time slots may contain objects of the types 'meeting', 'holiday', and 'business travel', and time slots may be located in 'month calendar', 'week calendar' etc.). The actors/roles slot and the related competence slot are meant to indicate which actors are allowed which manipulations with the object (like creation, inspection, transport).

In our example two roles are indicated each with the related competence: meeting participants can initiate a meeting cancellation, and meeting initiators may prohibit cancellation.

Fig. 10 shows the template for a basic task (see Section 5.2.1). The slots have been filled in based on the design exercise for a calendar and meeting organizer system. The slot for task relations indicates which other tasks (related to other roles) are triggered by the current task, or, alternatively, are triggering the current task.

The example of Fig. 10 shows that cancellation of a meeting will be a trigger for a secretary's task to forward cancellation messages, or for the meeting participants for receiving a cancellation message. The slots for initial and final state indicate relevant

Basic task: Cancel meeting	
relevant objects: meeting m participation_list l cancellation message c timeslot t todo_list d	
actors/roles: meeting participant	
task relations: one of: - forward meeting cancellation (role: secretary) - receive meeting cancellation (role: member of participation_list)	
initial state: timeslot t blocked with meeting m	**final state:** timeslot t empty
precondition: meeting initiator agrees on cancellation	**postcondition:** delivery of cancellation message is confirmed
user actions: - initiate cancellation - indicate timeslot t	**events:** - send cancellation message c to all participants on participation_list l - copy message m to todo_list d - empty timeslot t
comments: Meeting initiator cancels meeting	

Fig. 10. Basic task template – example from the electronic calendar and meeting organizer.

characteristics of the situation for this task (in the example, the time slot is blocked, or empty, respectively). The slots for precondition and postcondition indicate conditions for starting and ending the task (the task in the example may only be performed if the meeting initiator agrees, and is considered to be completed only if delivery of all cancellation messages is confirmed). In the case of our example the user has to initiate the cancellation process, and has to specify the time slot concerned. In the slot 'events' the system events have to be specified as far as relevant for the user (i.e., the basic task definition is specifying aspects of the UVM).

This formalism is relatively easy to write and to read for humans (in any case more so than traditional HCI formalisms – for an overview see De Haan et al., 1991), and interfaces easily to software engineering tools and methods. It enables a certain amount of flexibility in formality that is useful in early phases of modeling and, on the other hand, may be readily supported by design tools for consistency checking and automatic translation into a grammar like ETAG (Tauber, 1990) or GOMS (Card et al., 1983) for analytic design activities and for prototype generation.

The concepts used in this object-oriented representation should be considered as a first attempt, based on the conceptual framework, on envisaged analytical design methods like ETAG, and on considerations about the relevance of distinguishing and grouping notions in relation to users' knowledge and group knowledge of the task world.

A major feature of the formalism is the use of templates for the description (and, in later design stages, the specification) of objects and tasks (including protocols and strategies) and their relation to actions, to actors and roles, and to situation aspects (conditions, initial and final states). These templates are provided with slots for the indication of internal structure and external relations, indications for the relation between task structures and roles, and slots for relations between tasks (e.g., mutual tasks in interaction: sending and receiving, asking and forwarding information). State of the art CASE environments should provide easy vehicles for defining templates and for using them with the help of syntax-sensitive editors, tools for monitoring semantic relations, and semi-automatic support to 'translate' transcription records into formalisms for analytic purposes.

7. Conclusions

Our contribution advocates the distinction between several modeling activities in the design of complex systems for end-users: task model 1, task model 2, and the UVM. A conceptual framework has been offered and an object-oriented formalism is suggested, to make design decisions explicit, to document them for backtracking and iterative design activities, and to communicate between members of a design team. In actual design exercises in several classes of students of general universities and polytechniques as well as in post academic courses on user interface design, we have applied GTA as method for the design of complex systems. GTA developed during these courses, and the current state of development is illustrated by examples from two recent design exercises that could successfully deal with task analysis by using our methods and formalisms. Fig. 11 shows an overview of the different design activities and the

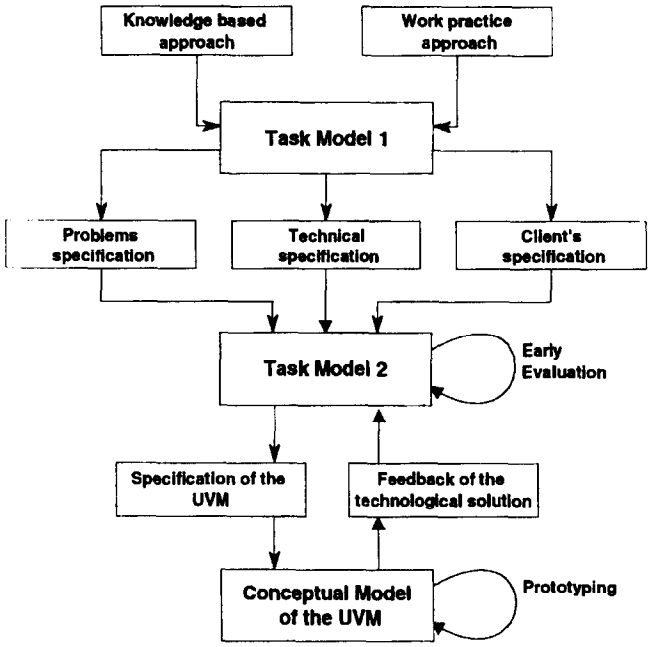


Fig. 11. GTA – design activities and model types in the early phases of the design of groupware.

specification of models during the task analysis phases of the design of complex groupware.

The combined methods of collecting information on the current task situation, the construction of task model 1, and the negotiations (in the design team and with the client) on task model 2 and the subsequent elaboration of the UVM turned out to be learnable, and useful in practice. Design students spent a few days on learning the methods, and collaborated in design teams of multi-disciplinary nature in specifying the different models and negotiation between various design activities using the formalisms suggested above. Analytic methods, hermeneutics, and ethnographic methods showed to enable a united formal modeling approach that may facilitate transition of knowledge of current or actual task situations to the specification and design of new task situations. The conceptual framework and the object-oriented formalisms developed in the previous sections are the core of this approach.

Acknowledgements

We would like to thank Hillary and Peter Johnson, Brigitte Jordan, Bonnie Nardi, Dominique Scapin, Susanne Sebillotte, and Michael Tauber, who generously discussed task modeling and task knowledge acquisition methods and provided us with insight in their approaches and methods. Their ideas enabled us to draft this proposal to complement the different types of effort. The students in our classes contributed in teaching us

how to elaborate the ideas, how to communicate them and how to shape them into usable tools.

References

Card, S.K., T.P. Moran and A. Newell, 1983. The Psychology of Human-Computer Interaction. Hillsdale, NJ: Erlbaum.

De Haan, G., G.C. van der Veer and J.C. van Vliet, 1991. Formal modeling techniques in human-computer interaction. Acta Psychologica 78, 27–67.

Hirschheim, R. and H.K. Klein, 1989. Four paradigms of information systems development. Communications of the ACM 32(10), 1199–1216.

Johnson, P., 1989. 'Supporting system design by analyzing current task knowledge'. In: D. Diaper (ed.), Task analysis for human-computer interaction. Chichester: Ellis Horwood.

Johnson, P., H. Johnson, R. Waddington and A. Shouls, 1988. 'Task-related knowledge structures: Analysis, modeling and application'. In: D.M. Jones and R Winder (eds.), People and computers IV (pp. 35–62). Cambridge: Cambridge University Press.

Jordan, B., 1994. Ethnographic workplace studies and CSCW. Report IRL94-0026, Palo Alto, CA, IRL.

Nardi, B. (ed.), 1996. Context and consciousness: Activity theory and human computer interaction. Cambridge, MA: MIT Press.

Oberquelle, H., 1984. 'On models and modeling in human-computer co-operation'. In: G.C. van der Veer, M.J. Tauber, T.R.G. Green and P. Gorny (eds.), Readings on cognitive ergonomics – Mind and computers. Heidelberg: Springer-Verlag.

Payne, S.J. and T.R.G. Green, 1989. 'Task-action grammar: The model and its developments'. In: D. Diaper (ed.), Task analysis for human-computer interaction (pp. 75–105). Chichester: Ellis Horwood.

Scapin, D. and C. Pierret-Golbreich, 1989. 'Towards a method for task description: MAD'. In: L. Berlinguet and D. Berthelette (eds.), Work with display units 89. Amsterdam: Elsevier.

Sebillotte, S., 1988. Hierarchical planning as a method for task-analysis: The example of office task analysis. Behaviour and Information Technology 7(3), 275–293.

Tauber, M.J., 1988. 'On mental models and the user interface'. In: G.C. van der Veer, T.R.G. Green, J.-M. Hoc and D. Murray (eds.), Working with computers: Theory versus outcome. London: Academic Press.

Tauber, M.J., 1990. 'ETAG: Extended Task Action Grammar' – A language for the description of the user's task language'. In: D. Diaper, D. Gilmore, G. Cockton and B. Shackel (eds.), Proceedings INTERACT '90. Amsterdam: Elsevier.

Van der Veer, G.C., 1990. Human-computer interaction: Learning, individual differences, and design recommendations. Ph.D. dissertation, Amsterdam, Vrije Universiteit.

Van der Veer, G.C., J.C. van Vliet and B.F. Lenting, 1995. 'Designing complex systems – A structured activity'. In: Proceedings DIS '95, Symposium on Designing Interactive Systems (pp. 207–217), Ann Arbor, MI. New York: ACM Press.