Groupware Task Analysis

Gerrit C. van der Veer, Martijn van Welie

Department of Computer Science, Vrije Universiteit
de Boelelaan 1081a, 1081HV Amsterdam, The Netherlands
{martijn, gerrit}@cs.vu.nl
http://www.cs.vu.nl/~martijn/gta/

Abstract

Groupware Task Analysis is a task analysis method that deals with the context of use of a system in the broadest sense. The task world is seen from three viewpoints that deal with different aspects of the world. The processes of GTA and their background are described in detail. In addition a task analysis tool EUTERPE is described. EUTERPE is based on GTA and allows capturing of the task models and provides some basic analysis primitives.

1. INTRODUCTION

Analyzing a complex system means analyzing the world in which the system functions, or the "context of use", which comprises (according to standards like [1])
- the users;
- the tasks;
- the equipment (hardware, software, and materials);
- the social environment;
- the physical environment.

If we want to design systems for the context of use, we need to take these different aspects of the task world into consideration. In traditional literature on task analysis from the HCI (Human-Computer Interaction) mainstream, the focus is mostly on users, tasks, and software. Design approaches for GroupWare and CSCW (Computer Supported Collaborative Work), on the other hand, often focus on analyzing the world first of all from the point of view of the (physical and social) environment. In both cases, more recent developments at least include some aspects that belong to the other categories, but it still looks like one has to choose for either the one view or the other. Section 2 presents an idea of task analysis approaches from the classical HCI tradition, and, at the same time, provides our view on phases in task analysis. In Section 3 an ethnographic viewpoint, as frequently applied to the design of CSCW systems, is presented, where phases in the analysis process are hardly considered. As the result of combining approaches from both HCI and CSCW design, we developed our GTA (Groupware Task Analysis) framework of modeling task knowledge, which we will describe in section 4. Brigitte Jordan [2], though originally working from an ethnographic approach and focusing on GroupWare applications, provides a view on analyzing knowledge of the task world that is broad enough to cover most of the context of use as now defined by the above mentioned ISO document. We will illustrate Jordan's view in Section 5, distinguishing two factors:
- sources of knowledge: (1) individual knowledge, and (2) group;
- levels of communicability: (a) explicit, and (b) implicit.

Based on applying Jordan's 2 * 2 framework in actual design processes for large industrial and government interactive systems, and expanding the two factors from dichotomies to continuous dimensions, we describe a two-dimensional framework to analyze the different relevant sources of knowledge of the context of use. This framework provides a map of knowledge sources, that assists us to identify the different techniques that we might need in order to collect information and structure this into a model of the task world. Section 6 describe EUTERPE, a task analysis tool based on an ontology derived from GTA. The model used by the tool and how the resulting task models can be analyzed is described in section 7.

2. TASK ANALYSIS IN HCI DESIGN

Classical HCI features a variety of notions regarding task analysis. The concept is used to indicate different activities: (a) analyzing a "current" task situation, (b) envisioning a task situation for which information technology is to be designed, or (c) specifying the semantics of the information technology to be designed. Figure 1 gives an overview of the whole design process with all activities and sources of information.

Many HCI task analysis methods combine more than one of these activities and relate them to actual design stages (e.g., [3]). On the other hand, some authors do not bother about the distinction: GOMS (Card, Moran, and Newell,[4]) can be applied for any of them or a combination.

2.1 Analyzing 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 of to model this. Sebillotte (see [5]) elaborates a method to collect task knowledge and structure this into a hierarchical model of subtasks, Scapin and Pierret-Golbreich (in [6]) elaborate on this method and provide an object oriented formalism for modeling knowledge of existing task situations, like Sebillotte mainly focusing on activities. 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 (e.g.,[7]). Task model 1 is often considered of a generic nature (e.g., [5]), indicating the belief of authors in this field that different expert users have at their disposal basically the same task knowledge.

2.2 Envisioning 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. (see [3]) provide an example of a systematic approach where a second task model is explicitly defined in the course of design decisions. 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.

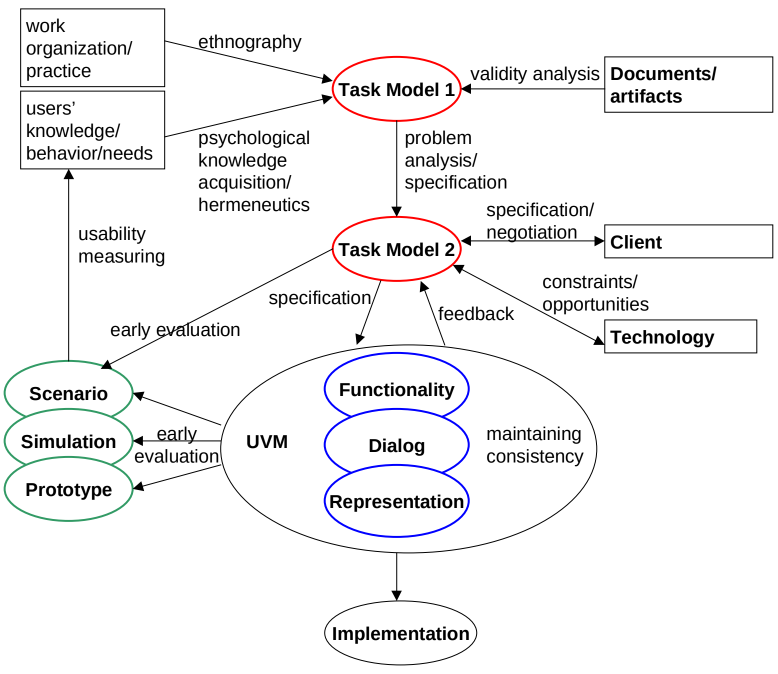


Figure 1 The design proces

2.3 Specifying technology (The user's virtual machine)

The third type of modeling activity that may be found in HCI design focuses on the technology to be designed. In principle this might be considered part of task model 2 (e.g., [4] in the case of GOMS). However, in other HCI approaches the actual design activities focus on the technology as such (e.g., [8]). In this part of design the activity is focussed on a detailed description of the system as far as it is of direct relevance to the end-user. Oberquelle (see [9]) introduces the concept "virtual machine" to indicate "the functionality of the system ... where implementation details and details of the underlying hardware are suppressed". Tauber ([10]) 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 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.

2.4 HCI task models represent a restricted point of view

All HCI task modeling is rather narrow focused, considering mainly individual people's tasks, although Johnson (e.g., [7]) considers the aspect of roles and the phenomenon of allocating subtasks to different actors. Most HCI approaches are based on cognitive psychology. Johnson refers to knowledge structures in long term memory. Tauber refers to "knowledge of competent users". HCI approaches focus on 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).

As a consequence of their source, HCI models seldom provide an insight in complex organizational aspects, in situational conditions for task performance, and in complex relations between tasks of individuals with different roles. Business processes and business goals are seldom part of the knowledge of individual workers, and, consequently, are seldom related to the goals and processes as found in HCI task modeling.

3. DESIGN APPROACHES FOR CSCW

CSCW work stresses the importance of situational aspects, group phenomena and organizational structure and procedures (Schael, [11]; Shapiro, [12]). Shapiro even goes as far as stating that HCI has failed in the case of task analysis for cooperative work situations, since generic individual knowledge of the total complex task domain does not exist. CSCW literature strongly advocates ethnographic methods.

3.1 Ethnography

Ethnographers 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 eyes 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 if CSCW: a design project). Ethnographers start their observation purposely without a conceptual framework regarding characteristics of task knowledge, but, instead, may choose to focus on activities, environments, people, or objects. The choice of focus is itself based on prior ethnographic observations, which illustrates the bootstrapping character of knowledge elicitation in ethno-methodology. Methods of data collection currently start with video recording of relevant phenomena (the relevance of which, again, can only be inferred from prior observation) followed by systematic transaction analysis, where inter-observer agreement serves to improve reliability of interpretation. 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.

3.2 The scope of ethnography

The ethnographic approach in unique in its attention to all relevant phenomena in the task domain that are not explicitly verbalizable by (all) experts (see [13]). The approach attends to 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.

Ethno-methodology covers the methods for information collection that might serve as a basis for developing task model 1 (and no more than this since ethno-methodology only covers information on the "current" state of a task domain). However, the methodology for the collection of data and the structuring 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, 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 on organizational and structural aspects of the work for which a system is to be (re)designed.

4. 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 (see [14]).

The framework as such is intended to structure task models 1 and 2, and, hence, as a guidance for choosing techniques for information collection 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 [14].

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 agents, work, and situation, respectively) that we will apply in our approach are a superset of the main focal points in the domain of HCI as well as CSCW. Both design fields consider agents (‘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 this is only sometimes, and then mostly implicitly, considered. In this section we will elaborate our conceptual framework.

4.1 Agents

The first aspect focuses on agents. “Agents” often indicates people, either individual or in groups. Agents are considered in relation to the task world, hence, we need to make a distinction between agents as acting individuals or systems, and the roles they play. Moreover, we need the concept of organization of agents. In situations where modern information technology is applied, actors will sometimes be non-human agents, or systems that comprise collaboration between human agents and machine agents.

4.1.1 Actor

This label mostly 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: (1) psychological characteristics like cognitive styles or spatial ability (see [15]); and (2) task related characteristics like expertise or knowledge of information technology.

4.1.2 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 behavior and by objects (white coat, stethoscope, and wig).

4.1.3 Organization

‘Organization’ refers to the relation between actors and roles in respect to task allocation. The organization describes the agent structure in the task domain. 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). Delegation and mandating responsibilities from one role to another is part of the organization, as is the way roles are allocated to actors. In organizational structure roles can be hierarchically related in several ways: a role can be a subtype of another role (a sales manager is a manager), or roles may be part of a role (a nurse is part of the company health department, which is part of the personnel division).

4.2 Work

Some approaches refer to goals as the unit of description of work (GOMS: [4]), but we prefer to focus on the structural as well as dynamic aspect of work, hence, we will take ‘task’ as the basic concept, and ‘goal’ as an attribute. The concepts of task and goal in most frameworks have either a many to one or a one to one relation – several tasks may have the same goal, and each task has exactly one goal. 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, [13]). We 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.

4.2.1 Task

Tasks can be identified at various levels of complexity. The unit level of tasks needs special attention. Payne and Green (in [16]) 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 (1) the lowest task level that people want to consider in referring to their work, the ‘unit task’ (Card, Moran, and Newell, [4]); and (2) 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, [8]). 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.

4.2.2 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 ([6]) are relevant. 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.

4.2.3 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.

4.2.4 Protocols

This concept indicates actual 'rules' as turn out 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.

4.2.5 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. Strategies will be role related.

4.3 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.

4.3.1 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, or signatures. Non-material objects as well as physical objects 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.

4.3.2 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, [8], 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.

4.3.3 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.

5. SOURCES OF KNOWLEDGE AND METHODS OF COLLECTING THE KNOWLEDGE
Collecting task knowledge for analyzing the current situation for a complex system has to start by identifying the relevant knowledge sources. In this respect, we refer to a framework derived from [2], see Figure 2.

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

Figure 2 Dimensions of knowledge of complex task domains

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 a model of the current task world.

5.1 Collecting task knowledge
Going from knowledge that is available from professionals in the task world and domain experts, via knowledge that is present in the culture and in the social environment towards artifacts and the physical environment, we encounter knowledge in the cognitive psychological sense, awareness and anecdotal material in the culture, and traces of manufacture and use as well as environmental opportunities and constraints.

Going from explicit knowledge, via skills and rule based behavior, through intuitive and instinct-like behavior in individuals and groups and culture, we meet documented knowledge and conscious representations, stories and myths, as well as unspoken and unspeakable insights that still prove to be valid for guiding or monitoring adequate behavior in the context of use.

For task knowledge in cell a, psychological methods will be used including those elaborated by [7] and [5]: interviews, questionnaires, think-aloud protocols, and (single person oriented) observations. For knowledge indicated in cell b observations of task behavior will have to be complemented by hermeneutic methods to interpret mental representations (see [15]). 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 (see [2]). 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 behavior, and turns out to ignore the situatedness of task behavior. Secondly, explicit group 'knowledge' (e.g., expressed in official rules and time schedules) often is in conflict with actual group behavior, 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.

It can be shown that different techniques of date collection and date analysis are needed for different types of knowledge, and these techniques seem to map systematically on to the "two-dimensional" framework of knowledge sources. Related to the different types of knowledge and the techniques is the notion of reliability of collection of information, and the validity of the resulting knowledge. We consider the validity of the knowledge in relation to the history and time aspects of the task world. E.g., experts may base their current knowledge on training they received in a different phase of equipment application, and documents may reflect a rule that is yet to be accepted by the authorities who control task performance.

6. EUTERPE, A TASK ANALYSIS TOOL

Our task analysis environment EUTERPE[1] has been developed to aid the process of task analysis. Although it is still under development, it is already being used in design projects in both educational and industrial settings. EUTERPE was developed because task analysis is still an activity that needs support. Task analysis is useful activity but it is often also a very unstructured and time-consuming activity. Many methods exist, but thoughts on task models and what they describe exactly have not been stabilized yet. Furthermore, task analysis methods usually only deal with task *modeling* and not really with task *analysis*. After the task world has been modeled it is up to the analysts to interpret the task model and find out where causes of problems can be found or where there is room for optimization of the work. These may be one of the reasons that cause task analysis to be both ineffective and inefficient.

A task model that can describe the task world including cooperative aspects and that allows some form of analysis could improve the task analysis process and outcome. Preferably the analysis of the task model should be done (semi-) automatically, thereby reducing the required effort of the analysts. In [17] a formal approach based on model checking techniques is described for analyzing user interfaces. A similar approach could also be applied to analyzing task models. However performing a formal analysis of a task model requires a formal representation of the task model that is suitable for doing an analysis, especially for analyzing cooperation. The task model therefore needs to be based on a task analysis theory that recognizes the cooperative aspects of the task world.

Although a formal analysis can be the basis for analysis it is not on the level analysts prefer to work. Hence representation tools can effectively hide the formalism and provide means to assist in analyzing the environment that is being studied. In addition, a tool can also provide more structured ways of doing task analysis. The next sections describe such a tool - EUTERPE - based on Groupware Task Analysis that supports formal analysis both on a logical and a visual level. Both the used models and the analysis primitives will be described in the next sections.

6.1 Theoretical background

During task analysis many aspects are modeled. Which aspects are modeled and which are not, is part of an ongoing discussion among scientist and practitioners. We have based our tool on an ontology which gives structure to the relevant aspects of the task world that we think are important. The next sections give a short overview of the ontology, for more information on the background of the ontology see [18].

6.1.1 A Task World Ontology

EUTERPE is based on a task world ontology that describes the way we look at the task world during task analysis. This ontology is derived from the conceptual framework of GTA that was described in the previous sections. It defines the relevant *concepts* and *relationships* between them that we regard relevant for the purpose of a task analysis. The ontology is of great importance because it is the conceptual basis of all information that is recorded and the way it is structured. Unfortunately, most task analysis methods do not define an ontology. Our ontology is derived from the three viewpoints from GTA and incorporates aspects of several other task analysis methods.

[1] EUTERPE is available at http://www.cs.vu.nl/~martijn/gta/

6.1.2 Concepts and Attributes

The *concepts* defined here are based on GTA and can be found in most other task models as well (with the exception of the *event* concept). This section will define the concepts and the next section will define their relationships in detail.

- **Object.** An object refers to a physical or non-physical entity. A non-physical entity could be anything ranging from messages, passwords or addresses to gestures and stories. Objects have attributes consisting of attribute-name and value pairs. What can be done with an object is specified by actions, for instance *move*, *change*, *turn off* etc. Furthermore, objects may be in a type hierarchy and can also be contained in other objects.
- **Agent.** An agent is an entity that is considered active. Usually agents are humans but groups of humans or software components may also be considered agents. Agents are not specific individuals (like "Chris") but always indicate classes of individuals with certain characteristics.
- **Role.** A role is a meaningful collection of tasks performed by one or more agents. The role is meaningful when it has a clear goal or when it distinguishes between groups of agents. A role is consequently *responsible* for the tasks that it encompasses and roles can be hierarchically composed.
- **Task.** A task is an activity performed by agents to reach a certain goal. A task typically changes something in the task world and requires some period of time to complete. Complex tasks can be decomposed into smaller subtasks. Tasks are executed in a certain order and the completion of one task can *trigger* the execution of one or more other tasks. A task could also be started because of an event that has occurred in the task world. Important for the task concept is the distinction between unit tasks and basic tasks, where (ideally) a unit task should only be executed by performing one or more basic tasks. The relationship between the unit task and basic task is interesting because it can indicate the problems that an agent may have in reaching his goals.
- **Event.** An event is a change in the state of the task world at a point in time. The change may reflect changes of attribute values of internal concepts such as Object, Task, Agent or Role or could reflect changes of external concepts such as the weather or electricity supply. Events influence the task execution sequence by *triggering* tasks. This model does not specify how the event is created or by whom.

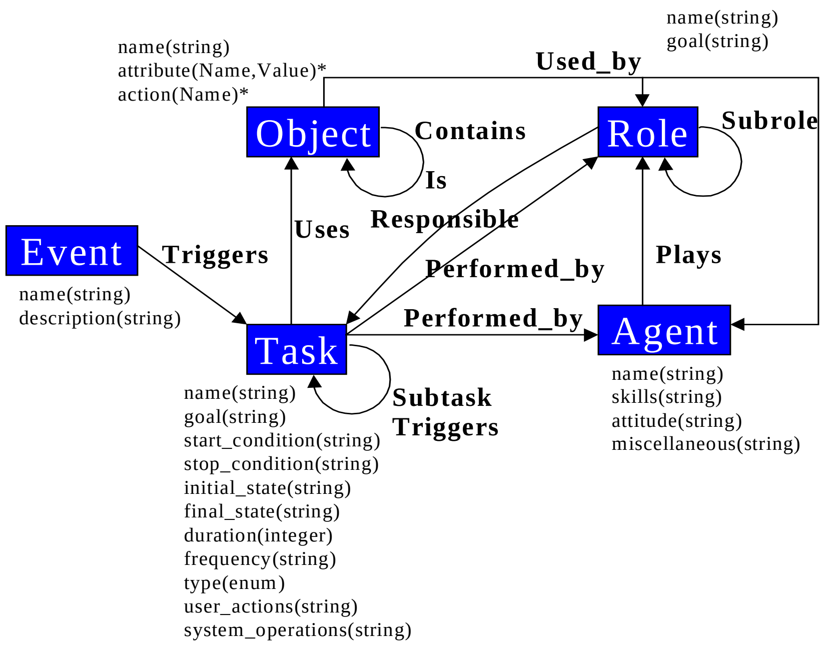


Figure 3 Concepts and relationships

6.1.3 Relationships

The concepts defined in the previous section are related in specific ways. In this section we sketch the relationships that we are using now. For each relationship the first-order predicate definition is given and explained. Figure 2 shows all the concepts and relationships together in a diagram.

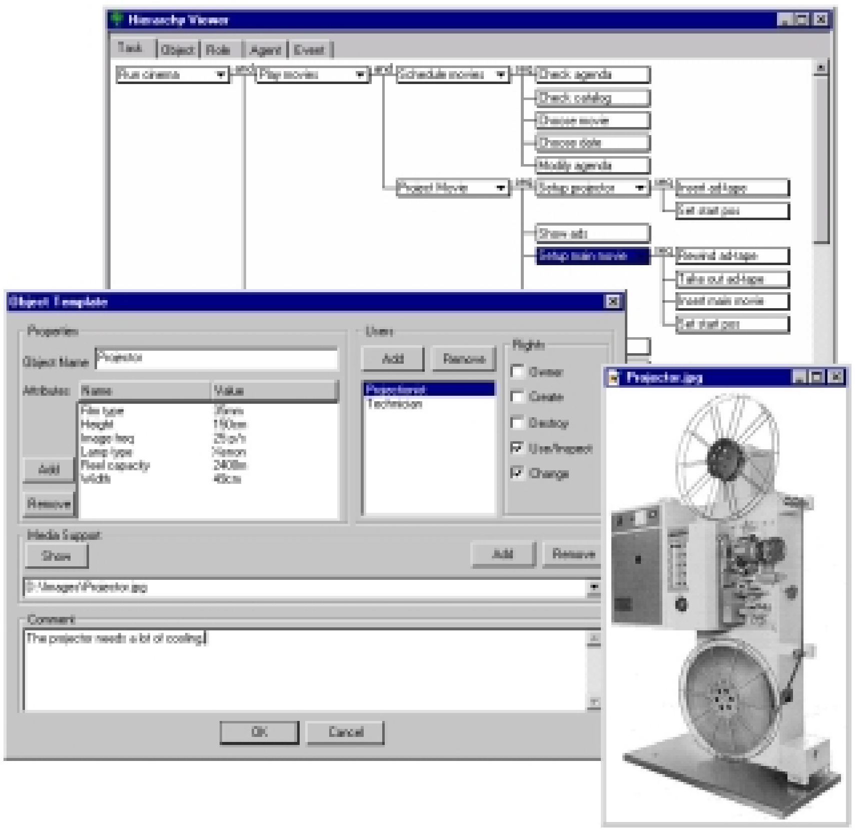
- **Uses.** The uses(Task,Object,Action) relationship specifies which object is used in executing the task and how it is used. The Action specifies what is being done with the object. It typically changes the state of the object.
- **Triggers.** The triggers(Task/Event, triggeredTask, triggerType) relationship is the basis for specifying task flow. It specifies that a task is triggered (started) by an event or a task and the type of the trigger. Several triggertypes are possible such as OR, AND, NEXT to express choice, parallelism or sequences of tasks.
- **Plays.** Every agent should play one or more roles. The plays(Agent, Role, Appointment) relationship also indicates how this role was obtained. Currently, the Appointment parameter can be ASSIGNED, DELEGATED, MANDATED or SOCIAL.

- **Performed_by.** The relationship performed_by(Task, Agent/Role) specifies that a task is performed by an agent. This does not mean that agent is also the one who is responsible for the task because this depends on his role and the way it was obtained. When it is not relevant to specify the agent that performs the task, a *role* can also be specified as the performing entity.
- **Subtask.** The subtask(Task, SubTask) relationship describes the task decomposition.
- **Subrole.** The subrole(Role, SubRole) relationship brings roles into a hierarchical structure. The subrole relationship states that a role includes other roles including the responsibility for the task that encompass the role. When a role has subroles the task responsibilities are added up for the role.
- **Responsible.** The responsible(Role, Task) relationship specifies a task for which the role is responsible.
- **Used_by.** The used_by(Object, Agent/Role, Right) relationship indicates who used which object and what the agent or role can do with it. The agents' rights regarding objects can be of existential nature (CREATE and DESTROY), indicate ownership (OWNER), or indicate daily handling of objects (USE, CHANGE).

Besides these relationships other relationships are used as well in EUTERPE but those have been expressed using the above relationships. These relationships provide information that is shown in certain representations such as templates. For instance for each concept a related(Concept) predicate has been defined which differs for each concept.

6.2 Deriving representations

The ontology only defines a structure for the task model data and does not limit or dictate any representation. The tool is based on a repository that contains the project data and all the representations are views on the repository. The task world ontology is specified in a logic programming language (Prolog) and is the main data structure for the repository. EUTERPE offers several different representations and all the representations are coherent because each representation is build up on the fly out of the same information specified using the ontology. For instance a task tree representation does not exist in the logical model but the structure is derived from the specified Subtask relationships of tasks. By issuing queries to the Prolog engine all the relationship can be inspected. Naturally EUTERPE allows most representations to be modified as well in which case the views need to assert the right facts in the Prolog engine. For instance when a new subtask is added a new fact *subtask(X,Y)* is asserted. This way the users of EUTERPE can work with the representations without having to deal with the logic representation underneath.



Object Name: Projector

Name	Value
Film type	35mm
Height	150cm
Image freq	25 p/s
Lamp type	Xenon
Reel capacity	2400m
Width	40cm

Users: Projectionist, Technician

Rights: ☐ Owner ☐ Create ☐ Destroy ☑ Use/Inspect ☑ Change

D:\Images\Projector.jpg

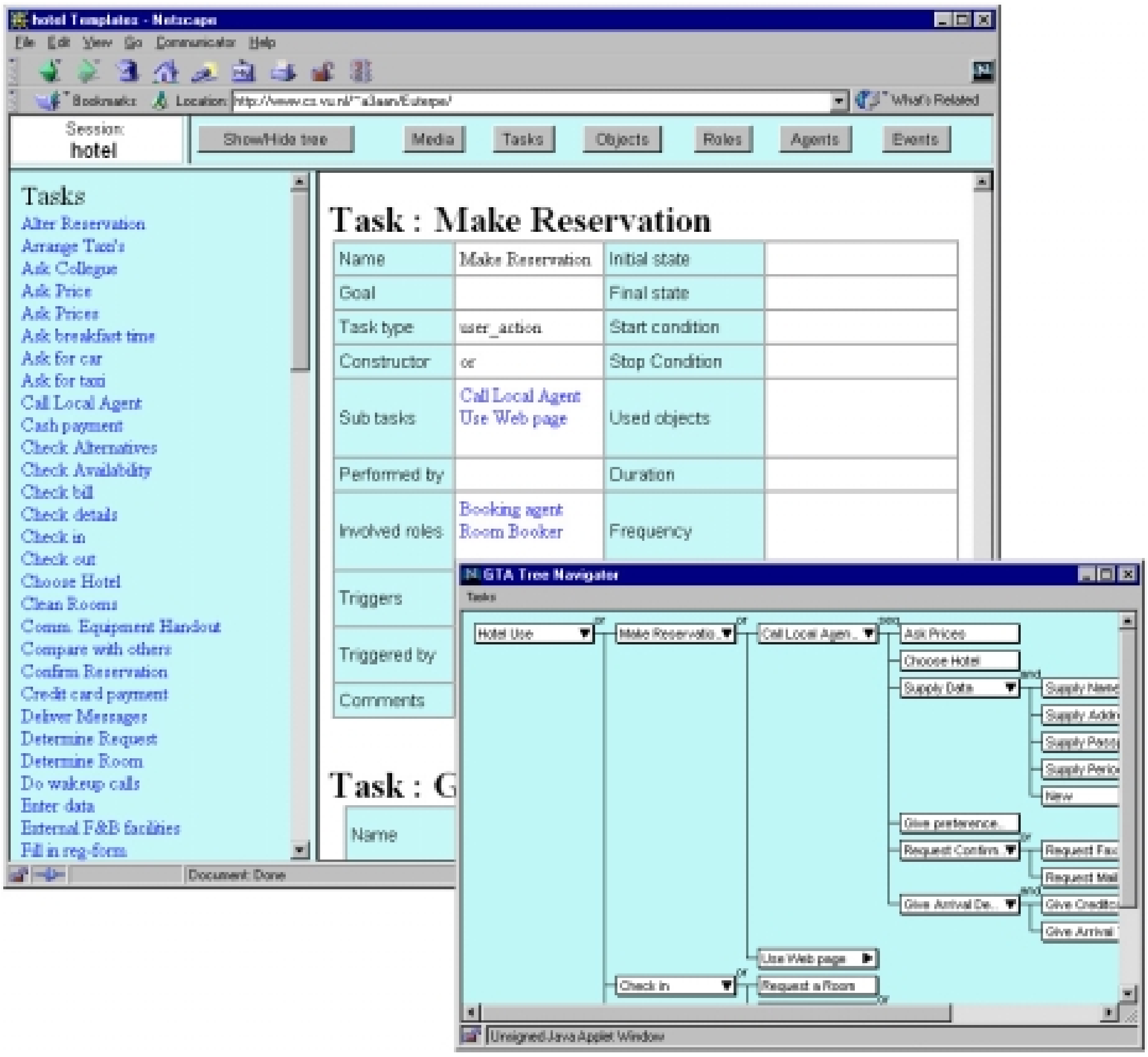
Comment: The projector needs a lot of cooling.

Figure 4 Some representations

Besides task trees EUTERPE also offers templates that show detailed task information and some context information such as the objects used in this task or the roles that are involved in this task or any of the subtasks. A somewhat different kind of view is the web browser view. At the moment we are working on a process flow view based on workflow representations.
Figure 4 shows some representations of EUTERPE.

6.3 Documenting a Task Analysis

EUTERPE has two ways of producing documentation. First of all by using the printing functionality. Task trees and object hierarchies as well as lists of events and agent can be printed. If a tree does not fit on one sheet of paper printing is automatically done tiled on multiple pages. The second way of producing documentation is by exporting the specification to HTML. EUTERPE can generate a set of web pages including an applet containing a task tree that allows online browsing of the task analysis results. As a result these pages are a "read-only" view of the model since changes are not propagated back to the Prolog engine. All the web pages together can be seen as a hyperlinked task analysis document because for each concept that referenced a hyperlink is added. For instance, a reference to an object used in a task becomes a link to the description of that object and vice versa. Links to images and video fragments are also generated. Figure 5 show the HTML output. For navigation purposes and for getting a better overview a simple Java applet is also included. The applet shows the trees graphically and when a node is selected the browser jumps to the corresponding entity. In large design teams the produced HTML documents were put on a web-server and these constituted the main reference document for the other members of the design team.



Tasks
Alter Reservation
Arrange Taxi's
Ask Collegue
Ask Price
Ask Prices
Ask breakfast time
Ask for car
Ask for taxi
Call Local Agent
Cash payment
Check Alternatives
Check Availability
Check bill
Check details
Check in
Check out
Choose Hotel
Clean Rooms
Comm. Equipment Handout
Compare with others
Confirm Reservation
Credit card payment
Deliver Messages
Determine Request
Determine Room
Do wakeup calls
Enter data
External F&B facilities
Fill in reg-form

Task : Make Reservation

Name	Make Reservation	Initial state	
Goal		Final state	
Task type	user_action	Start condition	
Constructor	or	Stop Condition	
Sub tasks	Call Local Agent Use Web page	Used objects	
Performed by		Duration	
Involved roles	Booking agent Room Booker	Frequency	
Triggers			
Triggered by			
Comments			

Task : G

Name	

Figure 5 HTML documentation with tree applet

6.4 Multimedia Documents

As mentioned in the requirements the outcome of task analysis could be much more than text documents with some graphical representations such as trees. EUTERPE offers support for images, sounds and videos. Figure 6 shows a video fragment of a guest arriving at the check-in desk of a hotel. Each concept instantiation can have of list of media objects connected to it. For instance a certain object specification can have several images associated with it. Also scanned documents or video fragments can be added to clarify certain tasks. We found that especially video fragments turned out to be very useful because they are very effective in showing other members of the design team who were not involved in the task analysis how the task world looks like. Video recordings resulting from ethnographic study are scanned and broken up in short video clips that show some particular "hot spot". Such video clips are typically between 1 and 3 minutes long and conversion to MPEG 1 format which makes them suitable for playback without any special hardware support. When HTML pages are created the media files are optionally included so that they can be viewed online as well.

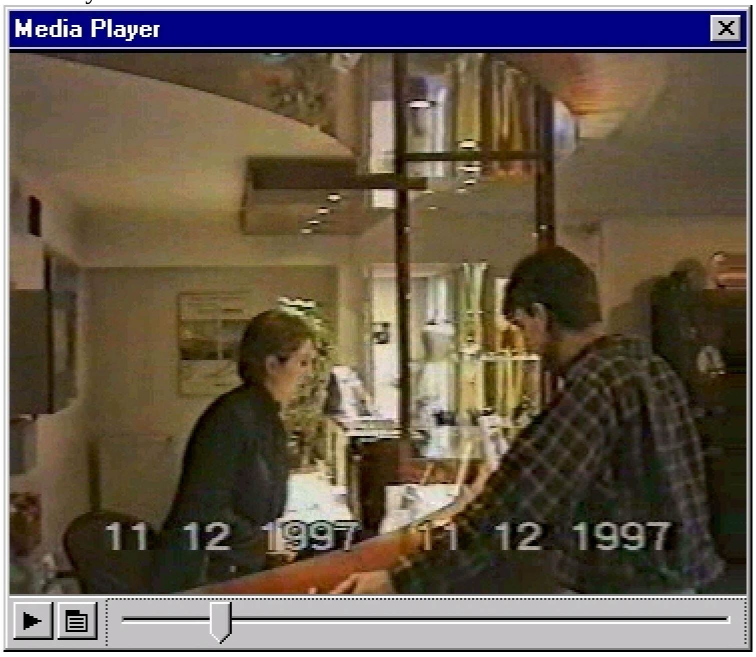


Figure 6 Example of a video fragment

7. ANALYZING A TASK MODEL

The representation of an ontology in logic allows us to analyze the task world in all its facets, the people with their work and the organization they are part of. One criticisms on task analysis has always been the fact that it remained unclear what exactly to do with the data: "we have the data, now what?" What should be next is an analysis of the data, finding problem areas and designing a "New World" that relieves these problems. The analysis that is usually done has an informal character and is based on insight on the data. However, we found that some problem areas have a more general nature and that they are domain independent.

- **Problems in individual task structures.** The task structure is sub-optimal because too many subtask needs to be done or certain tasks are too time-consuming or have a high frequency.
- **Differences between the formal and actual task performance.** In cooperative environments, usually regulations and work practices exist which are documented, for instance as part of ISO9000 compliance. In reality tasks are mostly not performed exactly as is described on paper and that "a single way" of how the tasks are done does not exist. When persons in a cooperative environment think differently about what needs to be done, problems may arise.
- **Inefficient interaction in the organization.** Complex tasks usually have many people involved who need to communicate and interact for various reasons, such as for sharing knowledge about tasks or because of responsibilities for the tasks. This can be the cause for time-consuming tasks but also for irritation between interacting people.
- **Inconsistencies in tasks.** Tasks are defined but not performed by anyone or tasks are executed in contradictory sequences.
- **People are doing things they are not allowed to do**. In complex environments often people have to do tasks for which they did not get an official permission or they are using/changing objects they are not allowed to touch.

Of course not all of these problems can be automatically detected. However using our model for describing task world models many characteristics can be detected semi-automatically by providing the analyst with a set of analysis primitives. Analyzing a cooperative environment can be done when the data present in the model is transformed into qualitative information about the task world. EUTERPE basically has two primitives of qualitative analysis. First of all visually in graphical presentations. When the data has certain features, these can lead to modifications of the graphical representations. The second primitive is to analyze the data at a logic level by putting some constraints on the model. Constraints that cannot hold may show interesting features of the task world. These two primitives allow several ways of analyzing a task model. We distinguish four ways: *inspection*, *analysis*, *verification* and *validation*. In the next sections these ways of analysis will be elaborated and clarified with examples.

7.1 Inspection

Inspection means browsing through your data. A task model based on the ontology is a complex model. In projects done by designers the task models typically consists of about 100 tasks, 20 object, 15 roles, 10 events and 10 agents. This a lot of information that needs to be understood. Graphical representations in general show specific aspects of the data, for instance a tree shows the hierarchical structure of tasks. Other useful representations include flow graphs, interaction diagrams, templates and hyperlinked structures. EUTERPE offers several of these representations and provides a coherent and consistent view on the data.

Additionally a coloring mechanism can be used to tune the graphical representations e.g., the coloring of nodes in a tree can be used to analyze task/agent allocation. The user can specify a condition for coloring of a node, for instance "all tasks performed by Chris". The user can choose from a range of predefined conditions or specify the condition directly in logic. Conditions can be arbitrarily complicated and range from showing task/agent allocation to showing instances of delegation of task responsibility. Figure 7 shows an example of a task tree with colored nodes. Another possibility is browsing through the concepts and seeing their details and relationships for instance by following links in the HTML representation or viewing templates.

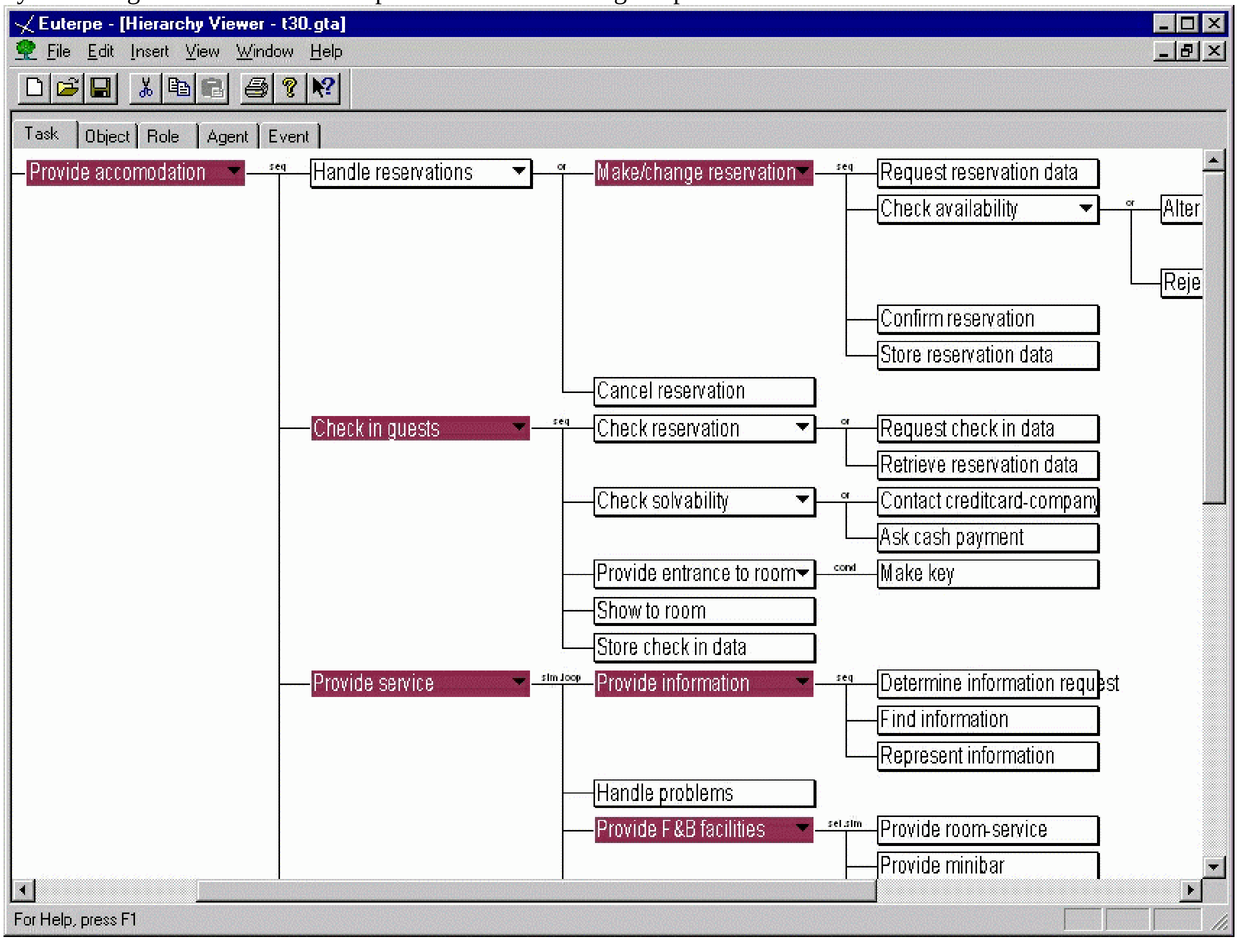


Figure 7 A tree with colored nodes

7.2 Analysis

Whereas inspection is merely "looking at" analysis is "finding out what is going on". Here the goal is to gain understanding of the task world and to find the nature and causes of problems. This can be achieved by using several different representations like those used in inspection and by using certain derived characteristics. For instance, coloring all tasks in which a certain role is involved may help to gain insight in the involvement of a role in the task structures. EUTERPE has built-in characteristics that can be checked on request but for the advanced analyst it is also allowed to specify additional characteristics. Some examples of predefined characteristics are:

- `agent X:`
  `all tasks performed by agent X`
- `cooperative task:`
  `all tasks where more than 3 agents are involved`
- `boring_task:`
  `all tasks that are performed more than 20 times per hour`
- `comlex_task:`
  `all tasks that have more than 3 levels of sub-tasks`

Cooperation can be seen as a dependency or interaction between certain tasks performed by different agents. Using that definition it is possible to define an expression that shows the frequency of interaction or how tight the cooperation is, for instance by counting the number of agents involved in a number of tasks. Because GTA looks at an organization of people and tasks instead of looking at one person, tasks are explicitly related to agents, objects and roles. All these relationships are established when the data is entered in EUTERPE.

7.3 Verification

This kind of analysis is on a more logical level. Verification concerns only the model as it has been specified. Only a limited degree of verification of a task model can be supported due to the inherently lack of formal foundations for task models. There is not a model to verify the task models with. The task world ontology merely defines the concepts and relationships without any constraints. This was done deliberately to give the analyst as much freedom as possible to specify what they find during data gathering. There are however constraints that we would like to have satisfied independently of the specify domain that is being studied. For example we would like that for each task there is at least one responsible role and that each task is really being performed by an agent. These constraints can be specified as logical predicates and can be checked automatically. Examples are:

- `unauthorized task performance:`
  `a task that is performed by an agent who's role does not encompass the responsibility for the task.`
- `unperformed tasks:`
  `a task where no performing agent has been specified.`
- `unhandled event:`
  `an event where no task is being triggered.`
- `occurance of delegation:`
  `a task that is being performed by an agent other than one that is responsible for the task.`
- `conflicting task sequence:`
  `a sequence A triggers B triggers C and the sequence A triggers C triggers B`

These constraints are similar to the characteristics used in analysis. However, they have been defined irrespectively to the specific study being done. They should hold in any domain. A task model were all constraints are obeyed may be considered "better" than one that does not obey all the constraints. In other words the constraints allow us to denote classes of model which may have an order of preference.

7.4 Validation

Validation of task models means checking if the task models corresponds with the task world it describes. In the process of validation one may find that certain tasks are missing or there are more conditions that are involved in executing a task. Often one finds that there are exceptions that had not been found in earlier knowledge elicitation. Consequently validation needs to be done in cooperation with experts of the task domain and cannot be done automatically by a tool. However it is possible to assist in the validation process for instance by generating scenarios automatically that can be used to confront the person from the task world. Such generated scenarios are in fact simulations of pieces of the task model. Generating simulations has not been implemented in EUTERPE but recent work on early task model simulations[19] has shown promising examples of early simulations based on task models.

7.5 Managing Conflicts

When doing an analysis of any of these types, problems or conflicts may arise. By conflicts we mean situations that need to be handled by the analyst. Examples of conflicts are contradictory data gathered from different persons describing the same task. In this case one of the persons may have made a mistake or forgot something. Another possibility is that different persons really do the task in a different way in which case it is very interesting to note this fact. Conflicts do not always need to be "solved" but they certainly require attention and should be seen as a hint for possible interesting aspect of the task world. The ontology we use allows inconsistencies and others causes of conflicts to be specified because we found in practice that it is often very important to be aware of these conflicts. Many analysts or designers develop models of a task world with the goal of finding *one* model that captures how the task world works. When modeling complex cooperative environments this is almost never the case and the analysts should not have this one model as the most important goal.

8. CONCLUSIONS

This paper gives an overview of Groupware Task Analysis from both a theoretical and practical perspective. The conceptual framework of GTA with the three viewpoints agents, work and situation, extends the classical task analysis approaches and makes a clear link to CSCW approaches. From a practical perspective a task world ontology was described that gives structure to the task models themselves and it was shown how the ontology is used in practice with our tool EUTERPE. Task analysis includes both modeling and analysis activities usually with an emphasis on modeling. A semi-formal approach based on the ontology allows various ways of analysis that are supported in our tool.

9. REFERENCES

[1] Nigel Bevan (1994), ISO 9241-11 Ergonomic Requirements for Office Work With VDTs.

[2] Jordan , B. (1996), Ethnographic Workplace Studies and CSCW, in: D. Shapiro, M. J. Tauber and R. Traunmueller, The Design of Computer Supported Cooperative Work and Groupware Systems , North-Holland, Amsterdam.

[3] Johnson, P., Johnson, H., Waddington, R. and Shouls, A. (1988), Task-Related Knowledge Structures: Analysis, Modeling and Application, in: Jones, D. M. and Winder, R., People and Computers IV pp. 35-62, University Press, Cambridge.

[4] Card, S.K., Moran, T.P. and Newell, A. (1983), The Psychology of Human-Computer Interaction, Lawrence Erlbaum Ass, Hillsdale.

[5] Sebillotte, S. (1988), Hierarchical planning as a method for task-analysis: the example of office task analysis, Behaviour and Information Technology, vol 7(3), no. 275-293.

[6] Scapin, D. and Pierret-Golbreich, C. (1989), Towards a method for Task Description: MAD, in: Berlinguet, L. and Berthelette, D., Work With Display Units 89 , Elsevier, Amsterdam.

[7] Johnson, P. (1989), Supporting system design by analyzing current task knowledge, in: Diaper, D., Task Analysis for Human-Computer Interaction , Ellis Horwood, Chichester .

[8] Tauber, M. J. (1990), ETAG: Extended Task Action Grammar - a language for the description of the user's task language, Proceedings of INTERACT '90, Amsterdam, Elsevier, Amsterdam.

[9] Oberquelle, H. (1984), On models and modeling in human-computer co-operation, in: van der Veer, G. C., Tauber, M. J., Green, T. R. G., and Gorny, P., Readings on Cognitive Ergonomics - Mind and Computers , Springer, Heidelberg.

[10] Tauber, M.J. (1988), On mental models and the user interface, in: van der Veer, G. C., Green, T. R. G., Hoc, J-M., and Murray, D., Working With Computers: Theory Versus Outcome , Academic Press, London.

[11] Schael, T. (1996), Information systems in public administration: from transaction processing to computer supported cooperative work, in: Shapiro, D., Tauber, M. J., and Traunmueller, R., The Design of Computer Supported Cooperative Work and Groupware Systems , North-Holland, Amsterdam.

[12] Shapiro, D. (1996), Ferrets in a sack? Ethnographic studies and task analysis in CSCW, in: Shapiro, D., Tauber, M. J., and Traunmueller, R., The Design of Computer Supported Cooperative Work and Groupware Systems , North-Holland, Amsterdam.

[13] Nardi, B. (1995), Context and Consciousness: Activity Theory and Human Computer Interaction, MIT Press, Cambridge MA, 1995

[14] Van der Veer, G. C., van Vliet, J. C., and Lenting, B. F. (1995), Designing complex systems - a structured activity, DIS'95, Symposium on Designing Interactive Systems, ACM Press, New York.

[15] Van der Veer, G.C. (1990) Human-Computer Interaction: Learning, Individual Differences, and Design Recommendations, Ph.D. Dissertation Vrije Universiteit, Amsterdam.

[16] Payne, S.J. and Green, T.R.G. (1989), Task-Action Grammar: the model and its developments, in: Diaper, D., Task Analysis for Human-Computer Interaction , Ellis Horwood, Cambridge MA.

[17] d'Ausburg, B. (1998), Using Model Checking for the Automatic Validation of User Interface Systems, Proceedings of DSV-IS98, Abingdon UK, Springer-Verlag, Wien.

[18] van Welie, M., van der Veer, G. C., and Eliëns, A. (1998), An Ontology for Task World Models, Proceedings of DSV-IS98, Abingdon UK, Springer-Verlag, Wien.

[19] Bomsdorf, B. and Swillus, G. Early Prototyping based on executable task models, CHI '96 Coference Companion, Canada, Vancouver.