MODELLING MULTI-USERS TASKS

F.Paternò, G.Ballardin, C.Mancini

CNUCE, C.N.R.

1. Introduction

Task models are the meeting point about multiple views of an interactive software application because their full development often require the involvement of many expertises: the designer, the expert of the application domain, the end user, the software developer and so on. The need for this possible multiple contributions is derived also for their possible multiple use:

- *To improve understanding of the application domain*, in this case designers develop task models corresponding to the existing application and the modelling work can stimulate the request of many clarifications whose need may not be perceived just reading textual descriptions or documents;
- *To support effective design*, in this case designers develop an envisioned task model of a new application incorporating requirements received (for example to use some specific technology or to overcome some limitations for end user that occur in current applications) and the various information contained in the new task model can support the design and implementation of a concrete user interface;
- *To perform usability evaluation*, to compare how users interact with an application and the related task model (Lecerof & Paternò 1999) can be useful to understand limitations of the current design, whose improvement may imply changes in the corresponding task model.

Despite of the increasing recognition of the importance of task models in the HCI area there is a lack of engineered tools supporting their use. This limitation is a consequence of the lack of structured methods and precise notations for task models. In this paper we introduce and discuss how we are solving such a problem by developing an automatic environment for task models specified in the ConcurTaskTrees notation (Paternò 1999).

1. Modelling Multi-Users Applications

An important aspect to consider is that with the increasing availability of network applications in all type of environments there is a natural tendency to develop applications where multiple users can remotely cooperate to reach common goals. It is thus important to be able to develop task models where it is easy to express relationships among activities performed by multiple users.

Moreover, even if the idea of using task models to support design and development of user interfaces for single user interfaces has been considered in various research proposals we can note that less attention has been paid to the case of multi-users applications. While even interesting research environment supporting development for multi-users applications (Roseman & Greenberg 1996) have solved problems related on how to support flexible internal mechanisms but they are rather difficult to use. We can find modelling approaches that consider multi-users application such as UML but they give rather poor support to the design of the user interface and their description of tasks has some limitations.

We use the ConcurTaskTrees notation to represent task models. It is a graphical notation where tasks are logically structured in a hierarchy and with a wide variety of operators that allow designers to describe flexible, dynamic and interactive environments. It uses different icons to indicate how the task performance is allocated (to the user or the application, or an interaction of both).

The notation has been designed so as to be intuitively understandable even by people who have not background in notations and to be powerful enough to express a rich set of possibilities. In this respect it has been an improvement with respect to previous practice because some of the previous approaches either considered only sequential tasks (Card et al. 1983) or tended to specify a lot of details that are not particularly important in designing task models.

We have also developed a set of criteria to support the concrete design and development of user interfaces taking into consideration various aspects of the task model and the information contained in it. For example, the temporal relationships among tasks are a source of various useful indications. They allow designers to identify the tasks that are logically available for performance at any time. This can be useful to identify when

activate the related interaction techniques thus contributing to define the dialogue part of the user interface. There are various levels of consistency that can be considered: a complete consistency can mean that only the interaction techniques associated with logically enabled tasks are perceivable and reacting. A weaker level of consistency is that there may be interaction techniques associated with tasks disabled that can be perceivable (but not reactive) to allow the user to understand their relationships with those reactive. A further possibility is to allow some interaction techniques associated with not enabled tasks to be reactive. This could allow users to change how to use the application. Even if this last option can easily generate the possibility to allow the user to make errors, actions not useful for the current tasks.

We have also developed a method supporting the possibility to obtain user interfaces consistent with the information contained in the task model. The first problem we have addressed is to identify the tasks that can be supported by the same presentation unit. To this end we have developed a tool that is able to take as input a ConcurTaskTrees specification and then to find the enabled task sets which are sets containing tasks that are enabled during the same period of time. Tasks belonging to the same set should be supported by the interaction and presentation techniques provided by the same presentation unit because they are logically enabled at the same time. Even if, as we have seen, this is not mandatory and designers in some cases may want to follow other rules.

We have then developed some further rules to identify the possible presentation units and also the specific interaction techniques associated with each task depending on its type and the type and cardinality of the objects that it manipulates.

The description of cooperative activities is currently done in ConcurTaskTrees by developing a task model for each user role involved and then developing a similar task model dedicated to what we call cooperative tasks, tasks that in order to be performed require activities by multiple users. In this part, the cooperative part we indicate the temporal relationships among such cooperative tasks and we decompose them until we reach tasks that are performed by only one user. These single user tasks are indicated in the cooperative part and then, if they are further refined, this decomposition is described in the task model of the corresponding role. This type of solution has various advantages: it is modular so that if designers are interested in the activities of a specific user they can consider only the related part as well as they have the possibility to focus on the cooperative aspects.

2. Tool support for task modelling

We have developed an approach (Paternò & Mancini 1999) supported by a tool (available at http://giove.cnuce.cnr.it/EL_TM.html) that allows designers in the initial phase when they have to develop their task model from scratch. We took into account that usually a lot of information is available in informal material, such as scenarios or use cases descriptions. Usually this information contains many indications of what tasks should be supported and the objects that they have to manipulate. Thus our tool supports their selection and their placement in a logical framework from where further development of the task model would be easier especially with respect to when designers have to start from scratch.

The ConcurTaskTrees notation allows designers to specify task models hierarchically structured with the possibility to indicate temporal relationships among tasks, objects that they manipulate, how their performance is allocated and so on. Besides, it is possible to specify cooperative applications where different roles are involved, for this purpose there is a part dedicated to the cooperative aspect that, using the same notation, allow designers to indicate relationships among activities performed by different users.

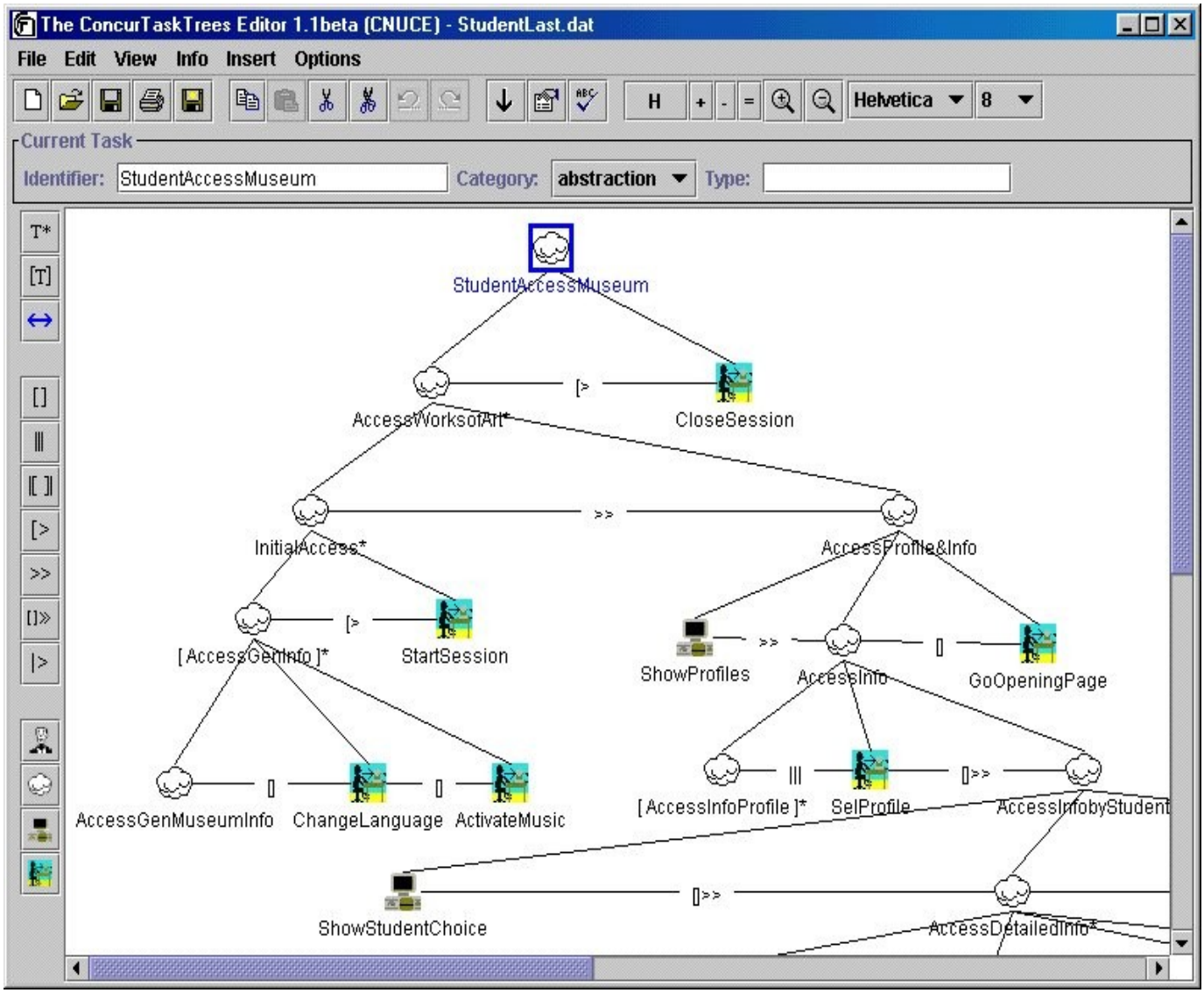

Figure 1: The editor of ConcurTaskTrees task models.

The editor (see Figure 1) supporting the development of ConcurTaskTrees task models (available at http://giove.cnuce.cnr.it/ctte.html) currently provides mainly three types of functionalities (we plan to introduce further features):

- *to allow designers to input the information requested by the notation*: tasks and their attributes (name, category, type, related objects, …), temporal relationships among tasks and other information;
- *to help designers in improving the layout of their specification*: supporting automatic lining up, movement of levels or subtrees of the task model, cut and paste of pieces of task model, folding and unfolding of subtrees, possibility to add tasks as brothers or as child of the current task and providing statistics concerning the task model (number of tasks by category, number of temporal relationships by operator, number of levels, …);
- *to check correctness of the specification*, for example it is possible to automatically detect whether temporal operators among tasks at the same level are missing or whether there is a task with only one child.

The tool also supports other functionalities such as inserting other task models in the current one or saving images containing the task model or part of it so that they can easily included in reports or documents. Besides it also includes a task model simulator.

1. Conclusions

In this paper we have discussed motivations and solutions for modelling multi-users tasks and the possible tool support that can be provided for this purpose. In the GUITARE project we are tailoring this approach for ERP applications, however it can be used also in other different application areas.

2. Acknowledgments

We gratefully acknowledge support from the GUITARE R&D Esprit Project (http://giove.cnuce.cnr.it/guitare.html).

3. References

Roseman M. & Greenberg S. (1996) Building Real-Time Groupware with GroupKit, A Gropware Toolkit*; ACM Transaction on Computer-Human Interaction*, Vol 3 N°1 March 1996; 66-106.

Card, S., Moran, T., Newell A. (1983) *The Psychology of Human-Computer Interaction*, Lawrence Erlbaum, Hillsdale, N.J., 1983

Lecerof A., Paternò F. (1988) Automatic Support for Usability Evaluation*, IEEE Transactions on Software Engineering*, October 1998, pp.863-888.

Paternò, F., *Model-Based Design and Evaluation of Interactive Applica*tion. Springer Verlag, 1999.

Paternò F., Mancini C. (1999) Developing Task Models from Informal Scenarios, *Late-Breaking Results ACM CHI'99*.