Tools for Task Modelling: Where we are, Where we are headed

Fabio Paternò
C.N.R. - CNUCE
Via G.Moruzzi, 1
56100 Pisa
+39 050 3153066
fabio.paterno@cnuce.cnr.it

ABSTRACT

Task models represent the intersection between user interface design and more systematic approaches by providing designers with the means to represent and manipulate abstractions of goal-oriented activities.

While task modelling and task-based design have long been considered, their adoption has been limited by the lack of tools supporting the development, interactive analysis and use of task models.

This paper discusses what support can be actually useful for designers and provides a taxonomy useful to compare tools for task modelling based on the experience accumulated with the CTTE tool. Some promising future developments will also be considered.

Keywords: Tools for Task Modelling, Interactive Simulation, Task Analysis.

INTRODUCTION AND MOTIVATION

One of the most important design principles to obtain usable interactive systems is "*Focus on the users and their tasks*". Indeed, of the relevant models in the human-computer interaction field, task models play an important role because they represent the logical activities that should support users in reaching their goals. Thus, knowing the tasks necessary to goal attainment is fundamental to the design process.

The need for modelling is most acutely felt when the design aims to support system implementation as well. If we gave developers only informal representations (such as scenarios or paper mock-ups), they would have to make many design decisions on their own, likely without the necessary background, to obtain a complete interactive system.

Task models describe the set of tasks supported by an interactive system and their relationships. Numerous task model formalisms and methodologies have been developed, including GOMS [9], UAN [12]; CTT [20], MAD [24], GTA [27], TKS [28]. However, one of the main problems in task modelling is that it is a time-consuming, sometimes discouraging process. To overcome such a limitation, interest has been increasing in the use of tool support. Indeed, tools developed for task models have been rather rudimentary, mainly research tools used only by the groups that developed them (see for example, Adept [28], IMAD* [10], DUKAS[1]). These tools where not mature enough to be used by other groups (even in research environments) and offered very limited features for task model analysis. More systematically engineered tool support is required in order to ease the development and analysis of task models and make them acceptable to a large number of designers. Some of the features needed include task modelling of cooperative and multi-user applications, which are on the rise, and the analysis of such models' contents, which can be complex in the case of many real applications.

This paper will provide a discussion of a set of functionalities that can be useful to support the analysis of task models. This discussion will be based on the experience garnered with the CTTE tool [16] (http://giove.cnuce.cnr.it/ctte.html). This tool supports a number of possibilities: representing concurrent tasks, cooperative tasks; calculating metrics, highlighting meaningful features of the task model and allowing comparison among different task models for the same interactive system; performing reachability analysis; interactive simulation of their dynamic behaviour; identifying relationships among scenarios and task models, and so on. Particular attention will be paid to those functionalities most useful to designers and evaluators of interactive systems.

Another important issue derived from the increasing availability of many interactive types of platforms (ranging from cellular phones to large screens) is how task models can support the design of multi-platform applications

1 http://www.cc.gatech.edu/gvu/user_interfaces/Mastermind/Dukas/index.html

(applications that can be accessed through a variety of interaction platforms).

More generally, a number of dimensions important for designers must be considered in any discussion of tools for task modelling:

?? How task models can be represented;

?? How tools can support the construction of the task model;

?? What metrics can be useful to analyse the models;

?? Whether and how interactive simulation can be useful for designers;

?? What other type of interactive analysis can be useful.

These aspects will be carefully considered also with the aid of a taxonomy useful for comparing task modelling tools based on experiences with the CTTE tool. Other tools developed for similar purposes will be positioned in this taxonomy.

The last part of the paper will be dedicated to discussing further functionalities that would be helpful in such tools, thus providing a research agenda for those interested in these topics.

HOW TASK MODELS CAN BE REPRESENTED

Task models can be represented at various abstraction levels. When designers want to specify only requirements regarding how activities should be performed, they consider only the main high-level tasks. On the other hand, when designers aim to provide precise design indications then the activities are represented at a small granularity, thus including aspects related to the dialogue model of a user interface (which defines how system and user actions can be sequenced).

Many proposals have been put forward to represent task models. Hierarchical task analysis has a long history and is still sometimes used. The concept of hierarchical decomposition of activities to describe has shown to be successful because it allows designers to consider the various possible abstraction levels while still maintaining a clear indication of the relationships among them.

More generally, such notations can vary according to various dimensions:

?? *syntax (textual vs graphical)*, there are notations that are mainly textual, such as UAN where there is a textual composition of tasks enhanced with tables associated with the basic tasks. GOMS is mainly textual, even if CPM-GOMS has a more graphical structure because it has been extended with PERT-charts that highlight the parallel activities. ConcurTaskTrees and GTA, are mainly graphical representations aimed at better highlighting the hierarchical structure. In ConcurTaskTrees the hierarchical structure is represented from top to down whereas in GTA it is from left to right.

?? *set of operators for task composition*, this is a point where there are substantial differences among the proposed notations. While GOMS supports only sequential tasks (with the exception of CPM-GOMS that also supports parallel tasks through the use of PERT-charts), UAN and CTT provide a much wider set of temporal relationships. This allows designers to describe more flexible ways to perform tasks.

?? *level of formality*, in some cases notations have been proposed without paying sufficient attention to defining the meaning of the operators. The result is that sometimes when task models are created, it is unclear what is actually being described. This is because the meaning of many instances of such composition operators is unclear.

HOW TOOLS CAN SUPPORT THE CONSTRUCTION OF THE TASK MODEL

Often it is difficult to create a model from scratch. To overcome this problem various approaches have been explored. CRITIQUE [14] is a tool that aims to create KLM/GOMS models from the analysis of logs of user sessions. The model is created following two types of rules: the types of KLM operators are identified according to the type of event, and new levels in the hierarchical structure are built when users begin working with a new object or when they change the input to the current object. In this approach the limitation is that the task model only reflects the past use of the system and not other potential uses.

U-Tel [26] analyses textual descriptions of the activities to support and then automatically associates tasks with verbs and objects with nouns. This approach provides some useful results, but it is too simple in order to obtain general results. The developers of ISODE [19] have considered the success of UML and provide some support to import Use Cases created by Rational Rose in their tool for task modelling. This environment also includes TAMOT a tool for modelling tasks specified with the DIANE+ notation.

In CTTE, to support the initial modelling work we give the possibility of loading an informal textual description of a scenario or a use case and interactively selecting the information of interest for the modelling work. In this way, the designer can first identify tasks, then create a logical hierarchical structure and finally complete the task model. The use of these features is optional: designers can start to create the model directly using our editor but such features can be useful to ease the modelling work.

To develop a task model from an informal textual description, designers first have to identify the different roles. Then, they can start to analyse the description of the scenario, trying to identify the main tasks that occur in the scenario's description and refer each task to a particular role. It is possible to specify the category of the task, in terms of performance allocation. In addition, a description of the task can be specified and the logical objects used and handled can be specified. Reviewing the scenario description, the designer can identify the different tasks and then adds them to the task list. This must be performed for each role in the application considered.

When each user's main tasks in the scenario have been identified, it might be necessary to make some slight modifications to the newly defined task list. This allows designers to avoid task repetition, refine the task names so as to make them more meaningful, and so on. Once designers have their list of activities to consider, they can start to create the hierarchical structure that describes the various levels of abstractions among tasks. The final hierarchical structure obtained will be the input for the main editor that allows the specification of the temporal relationships and the tasks' attributes and objects.

Another method supporting the automatic construction of task models of web sites is described in [18]. This method is able to take the HTML code of a web site and build the corresponding task model (or at least a part of it). It first analyses each page in order to identify the basic tasks supported, then it analyses the grouping techniques used in order to identify the corresponding higher level tasks. Next, it moves on to consider relationships among the Web pages in order to identify higher level tasks and their relationships. The result of the automatic reconstruction can still be edited by the designer for refining and completing the task model.

METRICS FOR TASK MODELS

The structure and the information of a task model can contain useful information for designers. It becomes important to identify relevant metrics that help them to capture such information and use it to analyse design solutions and compare them.

CTTE supports the identification of a number of metrics for the analysis of a task model. Such metrics can be used to compare task models. This can be useful when designers want to compare how people work in the current system and how they could work in a new envisioned system or are interested in comparing the implications at task level of two alternative designs. Previously no tool has given this support for task models. CTTE gives some information that can be helpful for this purpose. To be comparable the two task models should consider the same roles. The comparison is performed in terms of number of tasks, number of basic tasks (the tasks that are no longer decomposed), allocation of tasks, number of instances of temporal operators, structure of the task models (number of levels, maximum number of sibling tasks). This information can also be given for single task models in order to analyse them. By comparing this type of information it is possible to deduce some general feature of a solution with respect to another one. For example, a higher number of application tasks and a lower number of user tasks imply that there is a strong shift towards allocating task performance to the system, or a higher number of sequential operators implies that the solution supports a higher number of modes in its dialogues with the user.

The tool also supports comparison between models, designers have to select which part of the task model they want to compare and then the result of the comparison of the information associated with that part in the two task models appears. There is also a possibility of activating the presentation of the details related to some parameters. For example, if the details of interaction tasks are selected then the tool shows the interaction tasks of the selected role that are in one task model but not in the other and vice versa.

We would rather avoid having the tool provide definitive interpretations of these results, because they often depend on the type of application considered, and features that would characterise a good solution in one application domain may represent a bad solution in another. On the other hand, the automatic analysis highlights specific features of the solutions considered that otherwise would have been difficult to identify, especially when large models of real applications are considered.

In Euterpe [27] designers can specify constraints and heuristics. Constraints apply to every specification and should ideally have zero results. Heuristics can be used to analyse a specification, to find inconsistencies or problems. In this context examples of heuristics that can be queried on the specification are: what tasks involve a certain role, what tasks occur more than a certain number of times, what tasks have more than a certain number of subtasks, what tasks have more than a certain number of levels.

INTERACTIVE SIMULATORS

A simulator for task models can be useful to better analyse the dynamic behaviour of task models including those for cooperative applications. This feature is particularly meaningful when the notation used to represent the model allows the specification of many temporal relationships among tasks in addition to sequential tasks (such as disabling tasks, concurrent tasks, suspending tasks, and so on). This is a support that only a few tools provide (see for example VTMB [6]). Also in the case of tools for UML [8] this is a feature usually missing. In addition CTTE gives the possibility of simulating task models of applications where the resulting behaviour depends on the interactions of various users.

When analysing an existent application or designing a new one it can be rather difficult for the designer to understand the dynamic behaviour resulting from the temporal relationships specified in the task model. The reason is that, especially for real applications, the number of ways in which the application can evolve is high and it is difficult to mentally remember the various temporal constraints among tasks and their possible effects. It becomes important to support a what-if analysis aiming at identifying what tasks are logically enabled if one specific task is performed. To support this analysis of the dynamic behaviour of task models, interactive simulators can be helpful. The basic idea is that at any time they show the list of enabled tasks, according to the constraints specified in the task model. Before starting the simulation, the tool automatically checks that the task model is complete and consistent. When the simulator is started, then the window on the right displays the list of tasks enabled. Then, the user can interactively select a task to perform and the simulator shows what the next enabled tasks are. At any time, it is possible to go back through the performance of the tasks which means that the effect of the performance of the last task are undone and the list of enabled tasks becomes the same as that previous to the performance of the last task. At this point, the user can choose to go further backward in the task sequence or forward, either through the same path or a different one.

At any time, the designer can also display the specific sequence of tasks that has been performed in the current interactive simulation. They appear with an indication of the role of the user that performs the task. This is a way to interactively identify an abstract scenario that can be saved in a file and used to compare different task models. The tool is able to load a scenario created with another task model in order to simulate performance of the same sequence of tasks. If this is not possible, either because a task is not supported in the other model, or because the temporal relationships in that task model do not allow such a sequence, then it means that the scenario is not supported.

More generally, the simulator can be useful in several cases:

?? Designers can check whether the specified behaviour is really what they were looking to describe; this is important because, especially in case of large specifications, it is difficult to immediately understand the overall behaviour deriving from the combination of the hierarchical structure and the temporal operators;

?? It can support a multidisciplinary discussion where people with different background discuss design decision at the task level;

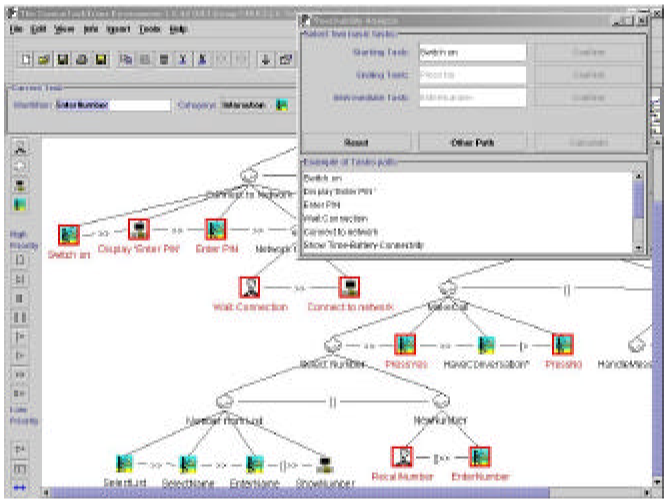
?? It can be employed as interactive documentation of an application to explain to end-users how to use it (indicating in which order tasks can be performed, possible choices and other dynamic information).

INTERACTIVE ANALYSIS

We have seen how an interactive simulator can be useful to analyse the behaviour of an interactive system. However, at times designers need other tools to analyse the content of a model. For example, once a model has been built it may not be immediate to understand how it is possible to reach a certain task after the performance of another one. To help designers to easily solve this problem we have added a new feature in CTTE that allows them to graphically select one starting task in the model and the desired task to reach. The tool is able to automatically calculate a sequence of basic tasks that shows an example of activities that should be performed to reach the desired goal (if any).

In models describing flexible ways to perform tasks there may be multiple paths connecting the two selected tasks. Thus, the tool allows the designer to ask for alternative solutions in the event that the first one is deemed to be uninteresting.

Another issue that can be interesting to address for designers is not only to indicate a starting task and an arriving task, but also an intermediate task that should be performed during the path connecting the two extremes. Figure 1 shows an example where the tool has identified a sequence of tasks that can be performed between the *Switch On* task and the *Press No* task, with the additional constraint that the path should include the *Enter Number* task. As you can see, the CTTE tool is able to provide one example of the possible results (if any) and also highlight it in the graphical representation of the model.



Switch on
Display 'Enter PIN'
Enter PIN
Wait Connection
Connect to network
Show Time-Battery Connectivity

Figure 1: An example of automatic task path identification.

TAXONOMY SUMMARY

Table 1 summarises some features deemed important for analysis tools of task models and shows how they are supported by some of the tools developed. It highlights how CTTE represents a useful contribution to understanding the possibilities in terms of the analyses that can be performed. The first row indicates whether the tools are able to consider concurrent tasks, that is, their analysis is not limited to sequential tasks. Next, we look at their ability to analyse cooperative tasks involving multiple users. Then, we consider whether they are able to calculate some metrics. We have seen that CTTE is also able to compare two models with respect to a set of metrics. Euterpe also supports the calculation of some metrics to analyse a specification, and help find inconsistencies or problems. The remaining features considered are the ability to predict task performance (this is usually supported by tools for GOMS such as QDGOMS [5]) and interactively simulate the task model's dynamic behaviour.

	CTTE	Euterpe	VTMB	QDGOMS	IMAD*	Adept
Concurrent tasks	XXX	XXX	XXX		XXX	XXX
Cooperative tasks	XXX	XXX				
Metrics	XXX	XXX				
Reachability	XXX					
Performance evaluation	XXX			XXX		
Simulator	XXX		XXX			

Table 1: Comparison of Tools for Task Modelling

MODEL-BASED USER INTERFACE DESIGN

We have seen that some design features can be analysed through an inspection of the task model. It is also possible to create an interactive application in such a way as to maintain direct correspondence between the task model and the user interface developed. Tool support for this transformation is possible. Here it is important to consider not only the tasks but also the objects that should be manipulated during their performance [7].

In particular, the temporal relationships among tasks can be used to structure the dialogues supported by the user interface, whereas knowing the basic tasks supported by each presentation (the set of information provided by a user interface perceivable by the user at a given time) is useful to select the most suitable presentation and interaction techniques

Various approaches have been proposed to derive concrete user interfaces from task models. A number of criteria can be identified for this purpose. For example:

?? the logical decomposition of tasks can be reflected in the presentation by explicitly grouping interaction techniques associated with tasks that share the same parent task.

?? sequential tasks can be supported in various modalities such as: separate presentations for each task rendered at different times; all the interaction techniques corresponding to the sequential tasks rendered in the same presentation (this is useful when the tasks are closely linked either because the sequence of tasks is performed multiple times iteratively or because the tasks exchange information); and lastly, through separate windows for each task (still active over the same period of time but which can be iconified if they take up too much space on the screen).

?? the type of task, the type of objects manipulated and their cardinality is another useful element. For example, if we have a selection task then we can immediately delimit the choice of the corresponding interaction technique to use to those supporting selection, in addition we can further limit the possibility of choice depending on the type of objects that can be selected, and, finally, if we know what the data cardinality is we can find only a few suitable interaction techniques, for example for choice among low cardinality values we can select a radio-button.

?? disabling tasks are tasks that interrupt other activities, in some cases to initiate new activities. They can be represented in the same manner (for example buttons with specific graphical attributes) and located in the same part of the layout (for example the right bottom part of the screen) for consistency.

Tools supporting user interface development starting with models are ADEPT, MOBI-D and Teallach. In Adept [28] a first set of guidelines was proposed for using task decomposition to guide the development of interface designs. They addressed how to reflect task decomposition in the design, how to use task actions and objects to determine the components that will actually appear, and the sequencing of tasks and corresponding dialogues structure.

In Mobi-D [23] the mapping issue is addressed according to three aspects: a mapping of domain objects with interactors according to some priorities; style attributes controlling some graphical and textual attributes; and strategy preferences indicating the preferred number of windows, the preferred way to implement sequential constraints, and the preferred interaction and navigation modalities.

Another tool has been developed in the Teallach project [3], where task models are considered and tool support focuses on the generation of the user interface for database applications.

MODEL-BASED DESIGN of CONTEXT-DEPENDENT SYSTEMS

One recent and important design issue is how to address the design of interactive systems accessible through both mobile and stationary platforms.

In general, it is important to understand what type of tasks can actually be performed in each platform. In a multi-platform application there are various possibilities:

- ?? Same task on multiple platforms in the same manner (there could be only a change of attributes of the user interface objects);
- ?? Same task on multiple platforms with performance differences:
  - o Different domain objects, for example, presenting information on works of arts can show different objects depending on the capability of the current device;
  - o Different user interface objects, for example, in a desktop application it is possible to support a choice among elements graphically represented, whereas the same choice in a wap phone is supported through a textual choice;
  - o Different task decomposition, for example, accessing a work of art through a desktop can support the possibility of accessing related information and further details not supported through a wap phone.
  - o Different temporal relationships, for example, a desktop system can support concurrent access to different pieces of information that could be accessed only sequentially through a platform with limited capabilities.
- ?? Tasks meaningful only on a single platform, for example, browsing detailed information makes sense only with a desktop system.
- ?? Dependencies among tasks performed on different platforms; for example, during a physical visit to a museum users can annotate works of art that should be shown first when they access the museum web site.

A method addressing such problems [21] is composed of a number of steps that allow designers to start with an overall envisioned task model of a nomadic application and then derive concrete and effective user interfaces for multiple devices:

- ?? *High-level task modelling of a multi-context application*. In this phase designers need to think about the logical activities that have to be supported and relationships among them. They develop a single model that addresses the various possible contexts of use and are able to indicate what platforms are suitable for each task.
- ?? *Developing the system task model for the different platforms considered*. Here designers have to filter the task model according to the target platform. This involves creating task models in which the tasks that cannot be supported in a given platform are removed and the navigational tasks deemed necessary to interact with the considered platform are added. Thus, we obtain the system task model for the platform considered. Such models are specified using the ConcurTaskTrees notation.
- ?? *From system task model to abstract user interface*. Here the goal is to obtain an abstract description of the user interface composed of a set of abstract presentations that are identified with the support of the enabled task sets and structured by means of various composition operators that reflect design principles. Then, still with the help of the task model, we identify the possible transitions among the user interface presentations considering the temporal relationships that the task model indicates.
- ?? *User interface generation*. In this phase we have the generation of the user interface. This phase is completely platform-dependent and has to consider the specific properties of the target device. For example, if the considered device is a cellular phone, such information is not sufficient as we also need to know the type of micro-browser supported and the number and the types of soft-keys available.

Another proposal that uses CTT and binary decision trees to describe task performance in different contexts of use is described in [22] [25].

TASK MODELS and USABILITY EVALUATION

Task models can also be helpful to perform usability evaluation. The most straightforward approach is the GOMS-like approach aiming to predict task performance on the basis of the knowledge of the time requested by each type of user interactions, the user and the system response time and of what actions are necessary to reach the considered goal [5].

This can be helpful to predict possible performances when comparing alternative designs. However, often the actual use of a system can bring up further design issues. Thus, interest has been increasing about how to use models for analysing actual user behaviour. WebRemUSINE [18] is an example of such approaches. It is an environment designed in order to perform intelligent analysis of Web browser logs using the information contained in the task model of the application. It performs an automatic evaluation of a Web site providing the evaluator with a set of measures, concerning also group of users, useful to identify usability problems derived from a lack of correspondence between how users perform tasks and the system task model.

The inputs for the tool are the task model and the log files recorded during the test sessions. The environment is mainly composed of three modules: the ConcurTaskTrees editor to specify task models in a hierarchical structure enriched with a number of flexible temporal relationships among such tasks (concurrency, enabling, disabling, suspend-resume, order-independence, optionality, …); the browser logging tool that has been implemented to record user interactions and that can be easily installed in a Web site; WebRemUSINE, the java tool able to perform an analysis of the files generated by the logging tool using the task model created with the CTTE tool. This approach supports remote usability evaluation of Web sites.

The evaluation performed provides information concerning both tasks and Web pages. These results allow the evaluator to analyse the usability of the Web site from both viewpoints, for example comparing the time to perform a task with that for loading the pages involved in such a performance. WebRemUSINE also shows an analysis of each log files.

WHAT NEXT?

We have seen how tools for task modelling have improved in recent years. However, there is always room for further improvements. Here is a list of issues that can form a research agenda for those interested in these topics:

Natural modelling, often the initial model is the result of brainstorming by either one single person or a group. Usually people start with some paper or whiteboard sketches. This seems an interesting application area for intelligent whiteboard systems [15] or augmented reality techniques able to detect and interpret the sketches and convert them into a format that can be edited and analysed by desktop tools.

Cooperative modelling, in some cases there are people who are located in separate environments (different offices in the same building, different towns, …) who want to discuss some issues regarding the model, with the possibility of highlighting potential problems and indicating possible solutions. Tools supporting these types of discussions can be helpful, though it would probably be best if they were based on some Web technology in order to guarantee portability and easy access.

Direct manipulation through user interface mappings. Models are abstract. In particular, they are useful to analyse potential dynamic behaviours, but often they would be more immediate if it was possible to map the tasks through some graphical or tangible representation of the elements manipulated during the task performance. In this way it would be possible to obtain a type of rapid prototyping environment to enable manipulation of the user interface, the corresponding model and the mappings between the user interface and task model elements. Changing the specification of the dynamic behaviour of the model would determine a change in the dynamic behaviour of the actual user interface. Pet-Shop [4] offers some of these features, but there is a need for environments easier to manipulate.

Use of information visualization techniques. In order to better analyse the content of task models it can be interesting to explore the use of information visualization techniques. Other related fields are benefiting from their application. For example, to aid analysis of the usability test data gathered, WebQuilt [13] provides filtering capabilities and semantic zooming, allowing the designer to understand the test results at the gestalt view of the entire graph, and then drill down to sub-paths and single pages. More generally, a promising area is to provide different interactive representations depending on the abstraction level of interest, or the aspects that designers want to analyse or the type of issues that they want to uncover.

Use of innovative non-WIMP techniques. In order to address the challenges of modern interactive systems with increasingly powerful applications and complex interfaces, new interaction techniques have been proposed, such as tool glasses and marking menus. The purpose is to obtain Post-WIMP interfaces able to find a better balance between power and simplicity also with the support of bi-manual manipulation. In [1] there is an example of application of these concepts to Coloured Petri Nets specifications. It would be interesting to explore their application to task model specifications as well.

CONCLUSIONS

In recent years interest in tools for task modelling has been increasing. This has generated the first engineered tools that are supporting the more widespread adoption of task models in the design cycle.

In this paper a taxonomy for analysing the features of these tools has been discussed. Then, the use of task models and related tools to support design and usability evaluation has been considered.

Lastly, a research agenda for those interested in this field has been proposed.

ACKNOWLEDGMENTS

This work has been supported by the CAMELEON IST R&D project (More information at http://giove.cnuce.cnr.it/cameleon.html).

I wish to thank Carmen Santoro and Giulio Mori for useful discussions on the topics of this paper.

REFERENCES

1. Beaudouin-Lafon M., Mackay E., Andersen P., at al., CPN/Tools: A Post-WIMP Interface for Editing and Simulating Coloured Petri Nets. Proceedings ICATPN 2001. pp.71-80, Springer Verlag LNCS N. 2075.

2. Baumeister L., John B., Byrne M., A Comparison of Tools for Building GOMS Models, *Proceedings CHI'2000*, ACM Press, pp.502-509, 2000.

3. Barclay P., Griffiths T., McKirfy J., Paton N., Cooper R., Kennedy J., The Teallach Tool: Using Models for Flexible User Interface Design, *Proceedings CADUI'99*, pp.139-158, Kluwer Academic Publisher.

4. Bastide, R., Navarre D., Palanque, P., A Model-based Tool for Interactive Prototyping of Highly Interactive Applications, Proceedings ACM CHI 2002, Extended Abstracts, pp.516-517.

5. Beard D., Smith D., Denelsbeck K., Quick and Dirty GOMS: A Case Study of Computed Tomography, *Human-Computer Interaction*, V.11, N.2, pp.157-180, 1996.

6. Biere M., Bomsdorf B., Szwillus G., The Visual Task Model Bulder, *Proceedings CADUI'99*, Kluwer Academic Publisher.
7. Bodart F., Hennerbert A., Leheureux J., Vanderdonckt J., A Model-based approach to Presentation: A Continuum from Task Analysis to Prototype, in *Proceedings DSV-IS'94*, Springer Verlag, pp.77-94.
8. Booch G., Rumbaugh J., Jacobson I., *Unified Modeling Language Reference Manual*, Addison Wesley, 1999
9. Card, S., Moran, T., Newell, A., *The Psychology of Human-Computer Interaction*, Lawrence Erlbaum, Hillsdale, 1983.
10. Gamboa Rodriguez F., Scapin D., Editing MAD* Task Description for Specifying User Interfaces at both Semantic and Presentation Levels, *Proceedings DSV-IS'97*, pp.193-208, Springer Verlag.
11. Gray, W., John, B., Atwood, M., "Project Ernestine: A Validation of GOMS for Prediction and Explanation of Real-World Task Performance", *Human-Computer Interaction*, 8, 3, pp. 207-209, 1992.
12. Hartson R., Gray P., "Temporal Aspects of Tasks in the User Action Notation", *Human Computer Interaction*, Vol.7, pp.1-45, 1992.
13. Hong J., Heer J., Waterson S., Landay J., ACM Transactions on Information Systems (TOIS) July 2001, Volume 19 Issue 3, pp.263-285.
14. Hudson S., John B., Knudsen K., Byrne M., "A Tool for Creating Predictive Performance Models from User Interface Demonstrations", *Proceedings UIST'99*, pp.93-102.
15. Landay J. and Myers B., "Sketching Interfaces: Toward More Human Interface Design." In IEEE Computer, 34(3), March 2001, pp. 56-64
16. Mori G., Paternò F., Santoro C., CTTE: Support for Developing and Analysing Task Models for Interactive System Design, to appear in *IEEE Transactions in Software Engineering*, September 2002.
17. Paganelli L., Paternò F., Automatic Reconstruction of the Underlying Interaction Design of Web Applications, Proceedings Software Engineering and Knowledge Engineering 2002. ACM Press.
18. Paganelli L., Paternò F, Intelligent Analysis of User Interactions with Web Applications, Proceedings of ACM IUI 2002, S. Francisco, pp.111-118, 2002
19. Paris, C., Tarby, J. & Vander Linden, K., (2001). A Flexible Environment for Building Task Models. *Proceedings of the IHM-HCI 2001*, Lille, France.
20. Paternò, F., *Model-Based Design and Evaluation of Interactive Application*. Springer Verlag, ISBN 1-85233-155-0, 1999.
21. Paternò F., Santoro C., One Model, Many Interfaces, Proceedings CADUI 2002, Kluwer Academics, pp.143-154.
22. Pribeanu C., Limbourg Q., Vanderdonckt J., Task Modelling for Context-Sensitive User Interfaces, Proceedings DSV-IS 2001, Sprinter Verlag, LNCS N. 2220, pp.49-68.
23. Puerta A., Eisenstein J., Towards a General Computational Framework for Model-based Interface Development Systems, *Proceedings ACM IUI'99*, pp.171-178.
24. Scapin D., Pierret-Golbreich C., Towards a Method for Task Description: MAD, *Work with Display Units* (89), pp. 371-380.
25. Souchon N., Limbourg Q., Vanderdonckt J., Task Modelling in Multiple Contexts of Use, PreProceedings of DSV-IS'2002, to appear.
26. Tam, R.C.-M., Maulsby, D., and Puerta, A., "U-TEL: A Tool for Eliciting User Task Models from Domain Experts", *Proceedings IUI'98*, ACM Press, 1998
27. van Welie M., van der Veer G.C., Eliëns A., "An Ontology for Task World Models", *Proceedings DSV-IS'98*, pp.57-70, Springer Verlag, 1998.
28. Wilson, S., Johnson, P., Kelly, C., Cunningham, J. and Markopoulos, P., "Beyond Hacking: A Model-based Approach to User Interface Design". *Proceedings HCI'93*. pp.40-48, Cambridge University Press, 1993.

.