Published in: Knut Nordby et al (eds.): Human-Computer Interaction, Proc. Interact '95. 5th Int'l. IFIP Conf. on Human Computer Interaction. Lillehammer, Norway, June 1995. London etc. 1995, Chapman&Hall. pp 297-304.

EXPOSE
HCI-COUNSELING FOR USER INTERFACE DESIGN

Peter Gorny
Carl v. Ossietzky University, Informatics Department, D-26111 Oldenburg
E-mail: Gorny@Informatik.Uni-Oldenburg.de Tel.: +49 441 798-2901 Fax: -2155

KEYWORDS: Computer-Aided Software Engineering, User-Interface Design, Knowledge-based Systems, Human-Computer Interaction, Software Ergonomics, Guidelines

ABSTRACT: *EXPOSE* is an expert system with the objective to provide user interface designers with empirical knowledge about common practice of the application domain, with software-ergonomic (HCI-) knowledge comprised from psychology and ergonomics research findings and with procedural knowledge about a principled approach (design rationale) to user interface design. The system provides the designer with a tool to describe the design decisions for a specific user interface subsequently in four models in order to reduce the design complexity: a conceptual model, a model of the dialog structure, a concretized model of the user interface and the prototype, following the four views of *MUSE* (Method for User Interface Engineering).

1. TYPE OF TOOL

EXPOSE (Expert System for Phase-Oriented Software Ergonomic counseling) is a decision support system consisting of five integrated parts (Fig. 1) to provide user interface designers with

- empirical knowledge about common practice in a specific application domain;
- knowledge-based information about decision alternatives, depending on the attributes of a given application system design context;
- tutorial information about Software Ergonomics and the principled design approach (design rationale) of the procedural model of *MUSE*[1] [Method for User Interface Engineering (Gorny et al., 1987; 1993; 1994)];
- help information about the usage of *EXPOSE*;
- a tool to describe the design decisions in a set of four models:

[1] No kin to the method MUSE (Long et al., 1992)

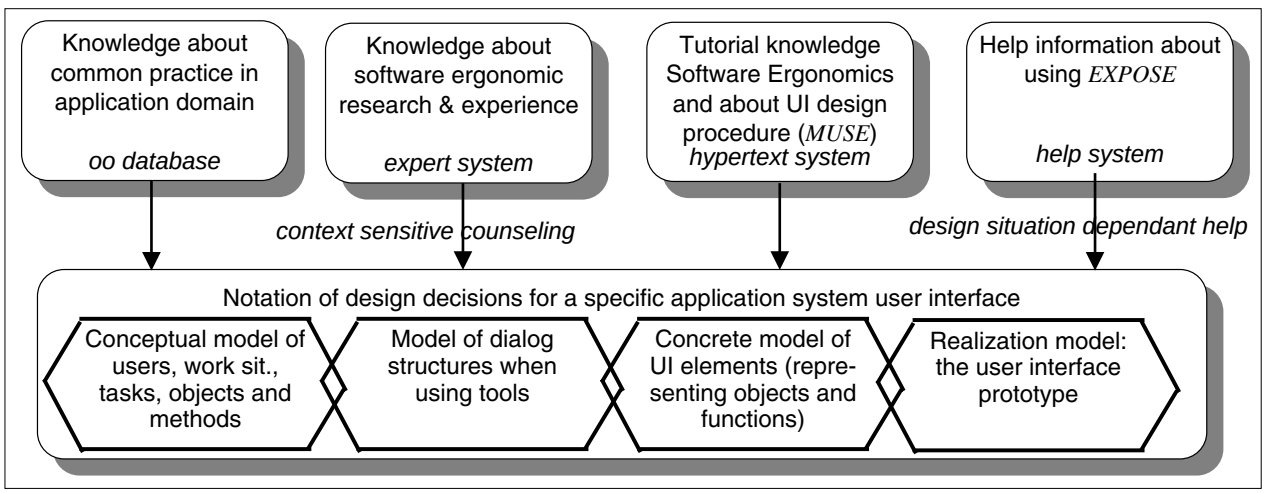

Fig. 1: The counseling components of *EXPOSE*

1. The conceptual model of the work environment and the task, for the objects and methods in the application area (abstracting from using the computer as a tool and as a represention for the objects to be manipulated.
2. The structural model for the dialog when using the application system.
3. The concrete model for user interface objects (representation of the work objects from model 1 and of the tool functions within the dialog structure of model 2)
4. The realization model: the interactive prototype of the user interface.

2. MOTIVATION

The method *MUSE* was developed to reduce the design complexity by ordering the decisions in four views. The corresponding tool *EXPOSE* system is meant to provide complementary counceling: it firstly offers a tool for noting design decisions, but it is not a specification tool for a complete application system functionality. Its use implies the design rationale of *MUSE* requiring a preceding object-oriented analysis [e.g., following OOA (Coad, & Yourdon, 1991)] of users, work situation and organisation, work objects and methods for manipulating the objects according to given tasks. Secondly it does not provide a tool for automatic user interface generation.

3. OTHER COMPARABLE APPROACHES

The described limited scope differs from the approaches used in various other projects, which aim at automatic generation of user interfaces, e.g., GENIUS (Jannsen et al 1993), JANUS (Balzert, 1994), or at providing knowledge from style guides and guidelines, e.g. INRA (Wandke & Hüttner, 1995), HyperSAM (Ianella, 1994) or SIERRA (Vanderdonckt, 1995), or at creating a software development environment with integrated advice and counseling, e.g. IDA (Reiterer, 1995) and DIADES II (Dilli & Hoffmann, 1994).

4. THE *EXPOSE* KNOWLEDGE DOMAINS

The knowledge base comprises four domains of knowledge (Fig. 1). Here we will only present those for application knowledge (common practice in an application domain) and for ergonomic knowledge (about software ergonomic research & experience).

4.1 Application Knowledge Base

This knowledge base presents the results of object-oriented analyses of application domains, e.g., office adminstration, CAD, or process control. Since it has its roots in the tool/material metaphor it centers on "**work objects**", which are commonly used in the application domain to solve certain tasks. Each object can be manipulated with related "**methods**". Each object method must be splitted into sets of "**tool functions**".

The designer can navigate through this knowledge base by following object inheritance hierarchies.

4.2 Software Ergonomic Knowledge Base

The knowledge base for software ergonomic knowledge is built as a rule-based expert system. Its rules are derived from investigations of HCI-problems from all releated areas (Psychology, Ergonomics, Computer Science, Industrial Design). Many of the findings are very dependant on their background and may not be

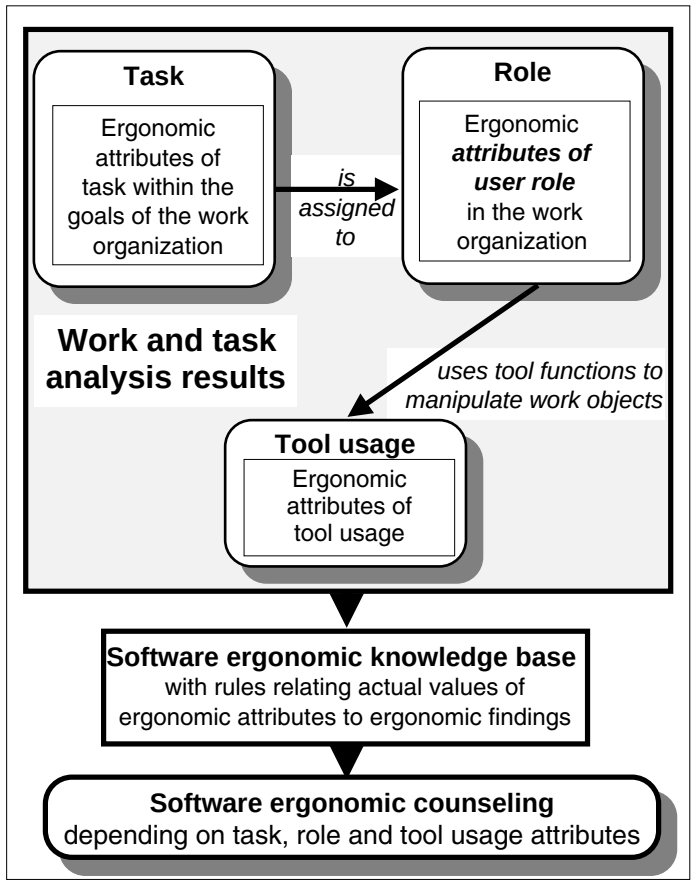

Fig. 2: Ergonomic advice produced by *EXPOSE*

generalized into sharp requirements. Therefore these requirements are linked to the ergonomic attributes of task and role (of the user in the given work organization) and the tool usage and map this tuple as rules into the expert system (Fig. 2). The rule will only be applied, when the attributes allow it. The systematics of the ergonomic attributes are derived from the findings of projects such as EVADIS (Oppermann, 1992), KABA (Dunckel, 1993) and TASK (Beck & Jannsen, 1993; Jannsen et al, 1993).

5. DESIGN PROCEDURE WITH *MUSE*

One of the main objectives of *MUSE* is to leave as many options open for the realization of the interface as long as possible and – contrary to "rapid" prototyping (Gorny et al., 1993) – to reduce the complexity of the design decisions by ordering the decisions into the four views mentioned above with a user interface prototype as a result. We presume that this principled prototyping approach will be handled – as the phase models of software engineering – with highly iterative walks through the views and with user participation.

5.1 Designing the conceptual UI model

In the first view the conceptual model of the interface is designed. Starting from a detailed specification of the task which the user has to perform within a given work environment (including an analysis of the work distribution and communication needs) the designer has to determine ergonomic attributes:

- task attributes (the quality and quantity of the wanted result and its predetermination, in respect to the task structure and the process variability etc.);
- user attributes (the degree of knowledge and experience in respect to the task and to tool usage etc.);
- tool usage attributes (the frequency, intensity, regularity of usage and the probability of interruption of the task by other tasks etc.).

Scenario: a tender submission system

As an example we present a task in the administration of a construction enterprise. One of the tasks is to submit tenders to customers on request or in response to a licitation. It consists of collecting all necessary technical information for the planned construction, linking it to cost factors, production facilities, technical and labor resources and aggregating this into a tender, which has to meet certain legal and commercial requirements. After taking other commercial information into account (e.g., the competition, the general market situation and the customer's solvency) the bid can be price-tagged and submitted to the customer.

Design Step 1.1: The designer selects an object, which is to be designed, the related task (within a hierarchy of more general tasks and goals) and the users ("roles"), who have to perform the task. Within this assumed scenario the designers could pick, for instance, the goal "submit a tender" with the tasks "check construction drafts", "calculate costs and availability of resources", "assemble cost factors", "draft work schedule", "check customer's solvency", "fill form", "send tender". For the task "fill form" the tender form is the central object and the designer decides to use it in his/her object-oriented approach and to identify the necessary methods for handling the form.

When asking for advice, the designer receives a list of objects commonly used in the application domain. Since the names of the objects will only seldomly correspond to the labels chosen by the designer, s/he has to match the object semantically with one from the presented list. Now the designer receives a list of manipulation methods which normally belong to the chosen common object (Tab. 1). When picking one of these common methods s/he is presented a list of common tool functions, which are necessary to realize the method in the computer system.

Design step 1.2: Determining the ergonomic attributes. The user of a tender system is typically trained in business administration with a solid knowledge of the engineering process. S/he may have only little knowledge and experience in computing, resp. in other computer applications. In spite of that s/he will probably – soon after an introductory period – request possibilities for customization of the program to meet the company's requirements and for individualization for her/his personal work style. The task requires great responsibility and initiative for the result from the user's side; the completion of the task can only be reached with a great deal of communication by several

object:	*tender form*;
related task:	*fill form* with tender data;
role:	*tender administrator*
object methods:	NewForm, DeleteForm, StoreForm, RetrieveForm, Fill_inForm etc.;

Table 1: Methods for the object "tender form"

media with the customer, with suppliers of construction material or special services, with engineers and administrators from the own enterprise. Frequent access to data bases for retrieval of technical and commercial data is necessary. The task is well structured; it offers a great variability with regard to the sequencing of the subtasks. The values for the usage characteristics can be summarized with frequent, but irregular occurance and intensive use. The frequency variation may reach from once a quarter to once a week. The collected ergonomic information is noted with the help of an editor for role and task descriptions (Fig. 3) and an editor for the tool usage descriptions.

Design Step 1.3: From the ergonomic attributes of task, role and tool usage we can derive the necessity of additional functionality complementing the application functions, which may have already been described in the system specification.

Staying within our scenario of tender submission the degree of freedom with respect to sequencing the subtasks leads to a *control function* in order to leave the default sequence. It would only be consequent, if the switching between subtasks was also possible without first completing the current one. The necessity of accessing many kinds of information resources while working on a subtask brings the designer to implement a control function for accessing the specific information server.

Equivalently the user needs *adapting functions* to move the presented information window to a suitable place on the display and change the window size, font, font size, cut and coloring, etc. The possibility that inexperienced users will have to solve the task is a reason to include some tutorial advice, also needed for users after a long interval without system usage: a *meta-function* . Similar meta-functions are needed to overcome the consequences of user errors, mistakes or slips: a system of error management is needed within the system. Other *control* and *adapting functions* are needed to cope with interruptions of the work process from outside (telephone calls, meetings, etc.). The

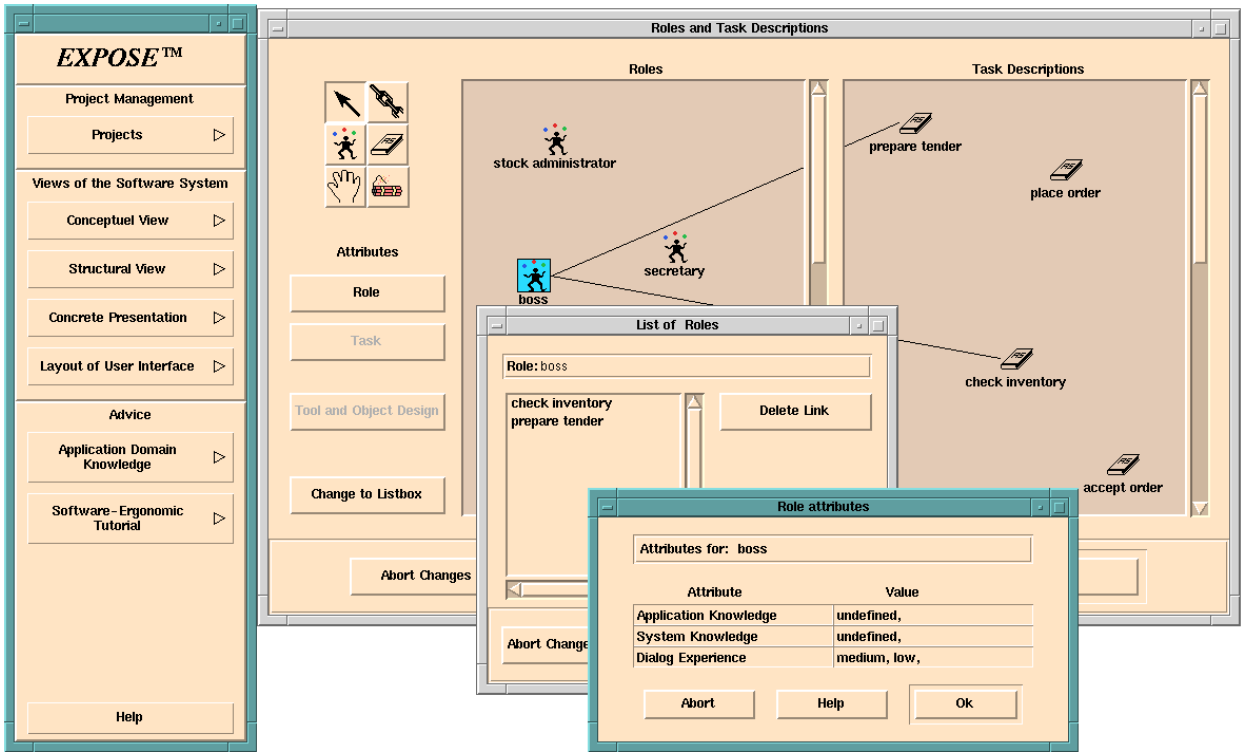

Fig. 3: Linking roles and tasks with ergonomic attributes

The designer creates icons for roles and tasks and labels them appropriately. Then he may link any of the tasks to each role by clicking on the chain icon after activating a task. By using the buttons under the headline "attributes", he can open a dialog panel to note the values of the ergonomic attributes for each role and task. By clicking on the role-task link line a dialog panel for noting the tool usage attribute values will be presented to him.

object *tender form* ;
common object with appropriate semantics (selected by designer): *form* ;
attributes of common object: *parts of form* ;
advice: *list of common methods for manipulating a form* : NewForm, DeleteForm, StoreForm, RetrieveForm, Fill_inForm etc.; *list of common tool functions* : e.g. **for method** *RetrieveForm* : OpenDirectory, ScrollDirectory, SelectForm_fromDirectory, OpenForm, etc. ; e.g. **for method** *Fill_in_Form* : Fill_inData, PositCursor_inOtherField, Erase_Data, Open_OtherFile, CopyData_fromOtherFile, Close_OtherFile, Browse_Directory, Change_FieldSize, Resize_Window, Novice_Help, Show_AlternativeActions, Interrupt_FormFilling, Resume_FormFilling, etc.
Additionally there will be tool functions to adapt the user interface (window size, font size etc.), to get help and to quit the action or the program etc.

Table 2: Advice about common practice: a list of common tool functions for the example "tender form"

task: *fill form with tender data*
ergonomic attributes of task: *low* predetermination of result quality, *medium* determination of result quantity, *well structured* result, etc.
role: *tender administrator*
ergonomic attributes of expected users: *high* application knowledge, *high* dialog experience, *low* system experience and knowledge etc.
tool usage: ergonomic attributes of tool usage: *high* frequency, *medium* regularity, *low* intensity, *high* interruptability etc.
object: *tender form* common object with appropriate semantics: *form*
attributes of common object: *parts of form*
selected common method: *Fill_in_Form*
ergonomic advice: *list of **evaluated** tool functions* for the selected common method: Fill_inData, **PositCursor_inOtherField**, Erase_Data, Open_OtherFile, Close_OtherFile, CopyData_fromOtherFile, **Browse_Directory**, **Change_FieldSize**, **Resize_Window**, **Interrupt_FormFilling**, **Resume_FormFilling**

Table 3: Ergonomic advice about evaluated tool functions (**Bold** functions are recommended because of ergonomic attributes.)

results of these assumed designer decisions are noted in Table 2.

Design Step 1.4: When now asking for ergonomic advice s/he will receive an *evaluated list of tool functions* - evaluated after necessity and usability criteria from the knowledge base (Tab. 3). The set of *necessary* tool functions for adapting the UI, for controling its use, and for fulfilling the task will vary depending on the values of attributes of the role, the task and the tool usage, the latter including frequency, regularity, intensity and interruptability.

Some functions are presented in bold only because of the specific ergonomic attributes of task, role and/or tool usage. The *tender administrator* is expected to have *high* dialog experience, the task performance is *regular* and *frequent* , etc., therefore s/he needs extra tool functions, e.g., for the possibility to change the sequence of filling data fields (PositCursor_inOtherField) or to resize a window , while a novice user would need functions not listed here, such as Show_AlternativeActions, Novice_ Help and would be disoriented by functions such as resizing a window or changing the size of an input field.

Design step 1.5: By accessing the HCI-knowledge base the designer is able to indicate the appropriate tool functions for further detailed design, and to transfer them to the design description (Fig. 4).

It is necessary to explain here, why such basic decisions have to be evaluated in a rather complicated manner: Firstly we have chosen a simple example to avoid long descriptions of the scenario. Secondly the example shows, that common practice has already lead to graphical user interface design tools with concealed design decisions, which in certain cases may not fit to actual ergonomic attributes. Thirdly the designer may

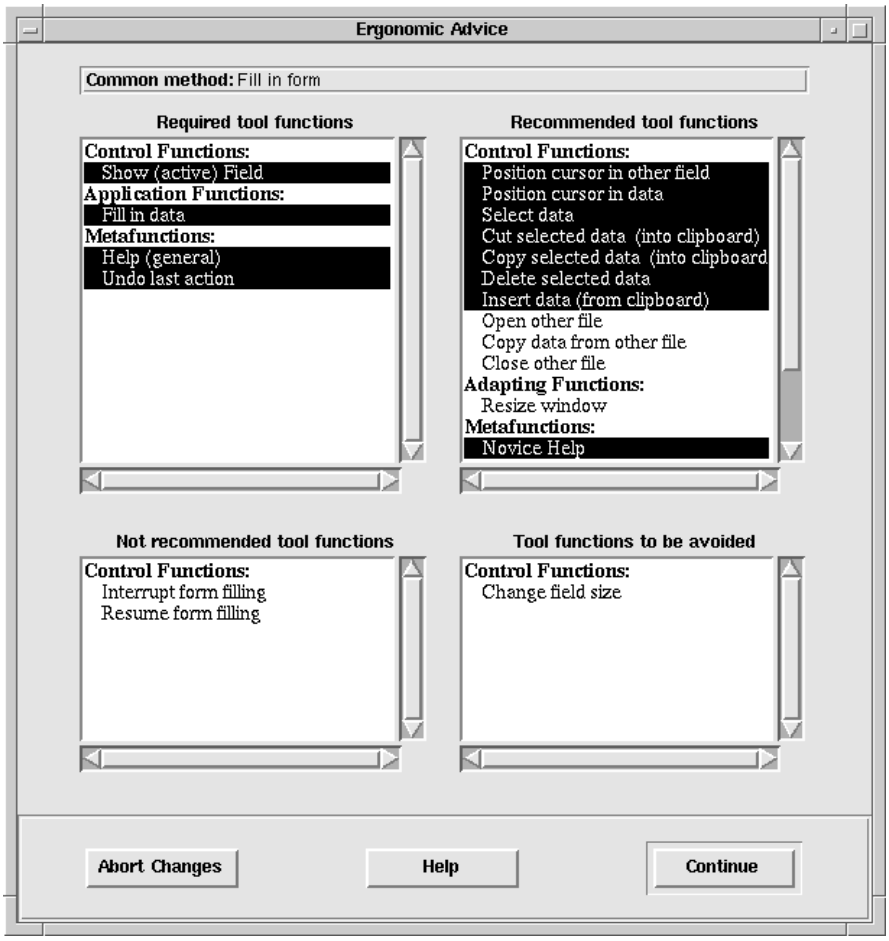

Fig. 4: Ergonomic advice for additional tool functions

often be in a situation, when s/he has to furnish tool functions "from scratch" normally included in window systems or user interface toolkits.

5.2 Designing the model of the dialog structure

This second view centers on design decisions with respect to dialog types and dialog structures.

Design step 2.1 : *Selecting dialog types* . For this purpose we distinguish between different *dialog types* :

- command dialog,
- data input dialog,
- multiple choice dialog,
- object manipulation,
- priority input (only in process control applications)

and combinations of these basic types to form complex dialog structures such as forms (masks), dialog panels etc. or combinations of direct manipulations of objects. The suitability of one of the types for a specific function depends on the attributes of task, role and tool usage.

Each of the dialog types can be implemented with different *interaction techniques* (the ISO draft standard 9241 Part 10 uses *dialog technique* here).

Design step 2.2 : *Combining sets of attributes.* When several users have to fulfill the same role and work on the same task, it may happen, that their tool usage differs (e.g., the tender administrator and his substitute). Then a different access to the tool functions might be needed. In order to avoid specific user interfaces for each of these cases, the interfaces can be combined to fulfill the usability needs of both user types: While the “normal user” may get access to the databases via a command input, because he normally knows the names of the files, the substitute needs the possibility to select the database in a multiple choice dialog.

Design step 2.3 : *The procedural model for the dialog.* The second group of design decisions in this view is related to the dialog process, which will be described as a procedural model. Fig. 5 shows the possible sequences for using the methods to fill a form. Not all sequences may be allowed for certain combinations of ergonomic attributes (e.g. for unexperiences users). A net editor supports the design decisions and allows interactive walk throughs in order to allow the testing of all possible pathes. As shown for "Get Form" in Fig. 5 a method can be refined for more detailed design of the tool functions.

5.3 Designing the concrete model of the dialog elements

Design step 3.1 : *Presentation of tool functions and objects.* In this third view the designer maps interaction techniques onto a physical medium: s/he decides to choose natural language for the textual input and/or output for a command dialog instead of using keyboard and display. The decision criteria for this advice have to be derived from special needs of users or from certain

conditions in the work environment, e.g., to keep the user's attention on a process (instead on the computer display), or to require the handling of objects (instead of the keyboard). Simpler recommendations involve the use of certain menu types or of interaction attributes for multiple choice dialogs. The recommendations for these design decisions given by *EXPOSE* have to be derived from Cognitive Science, from Gestalt- and from Perception Psychology, in combination with the ergonomic attributes.

The editor of this view supports a simplified presentation of all possible widgets. For a given design task it can be deduced from the data structure of the items (for instance, the customer data), that a "1:n-selection" is needed, and that the number of items may vary. This leads to the exclusion of check boxes and radio buttons, while menus, icon palettes and scroll lists still have to be considered. Since the number of items may grow large, *EXPOSE* will apply rules from HyperSAM (Iannella, 1994) and from Vanderdonckt (Vanderdonckt, 1993) about size and structure of menu selections and conclude, that a scroll list might be more suited.

Design step 3.2: *Presentation of data.* Also the presentation of data values on the display has to be decided in this view: single values, vectors, matrices, tables, forms, graphs, icons, photorealistic pictures, etc. Criteria for these decisions can be gained from the task analysis and from psychological considerations of affordances of objects. (Examples: knobs afford turning – buttons afford pushing – cords afford pulling – scroll bars afford … scrolling – slide controls afford value changes etc.)

The ergonomic knowledge base will take these mental models of possible basic actions into account when advising the use of certain widgets for the user interface. For example, the keyboard input of values representing measured physical properties and their output in a digital representation *suggest* a higher accuracy than normally given – a pseudo-analogue representation is more appropriate, a pseudo-analogue input via a slide control *is* already sufficiently inaccurate.

The concretization of an interaction technique includes the choice of the method for selecting. The methods vary from pointing (referencing) at objects on the screen via mouse or cursor keys to hitting keys, which are used as short labels for the menus items, or implementing pen writing or voice recognition systems.

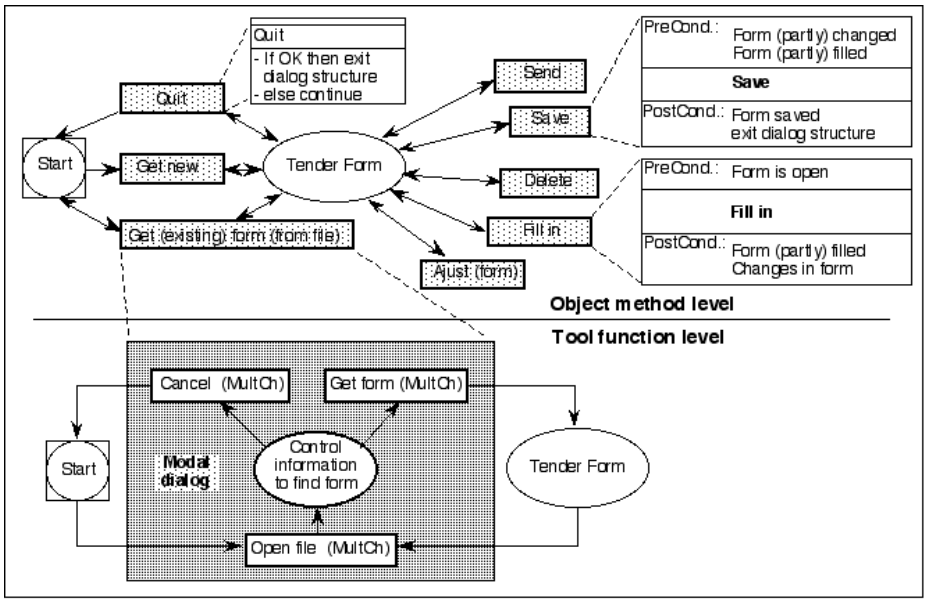

Fig. 5: Procedural model of dialog for the methods of "tender form":
This description allows an interactive incremental simulation of the dialog. An example: After "start" the user selects "Get new" [tender form]. Now there are eight existing continuations, but only five will be accessible, because sending or saving an empty form are excluded due to preconditions. After typing at least one letter into the form, it can be saved, but still not yet sent, since certain parts have to be complete. For the two methods "Save" and "Fill-in" the pre- and postconditions are listed in the box-blow-ups.

5.4 Designing the Prototype

The last view is used to develop a prototype of the user interface. *EXPOSE* can support this process by presenting (on demand) the recommendations for using graphical and/or acoustical attributes of elements to be displayed. This includes widgets, texts, icons (resp. hearcons), but it cannot, for example, give advice, how to find an appropriate vocabulary for the specific task.

6. COMPUTING PLATFORM

EXPOSE is implemented on a UNIX-workstation with OSF/Motif, the software ergonomic knowledge base the basis of NEXPERT Object, the user interface prototype is generated with the ISA dialog manager, and the tutorial hypertext system with MetaCard.

7. CONCLUSIONS

At the time of writing these lines the project has reached a state, where a prototype of the system can affirm the feasibility of the concept. Now the system can be validated by tests with software developers, since only the discussions with developers in a continous collaboration showed so far the need for counseling and approval for our approach.

In parallel the knowledge engineering process will continue in order to include more ergonomic knowledge and more practical knowledge about common practice in the office domain and in other application domains such as CAD and process control.

ACKNOWLEDGEMENTS

The project *EXPOSE* has been supported by a grant from the Federal Ministry of Research and Development, Bonn, under the Research Programme "Arbeit und Technik" (01 HK 190). It is a joint project of the Universities of Oldenburg and Rostock. This report came into being with support of the project coworkers Uli Daldrup, Ling Qin and Axel Viereck .

REFERENCES

H. Balzert(1994): Das JANUS-System. Informatik Forschungs und Entwicklung 9, 22-35.

A. Beck, Ch. Jannsen(1993): Vorgehen und Methoden für aufgaben- und benutzerangemessene Gestaltung von graphischen Benutzerschnittstellen. In: W. Coy, P. Gorny, I. Kopp and C. Skarpelis (eds.): Menschengerechte Software als Wettbewerbsfaktor. Stuttgart, B.G. Teubner. 200-221.

P. Coad, E. Yourdon (1991): Object-oriented Analysis. Englewood Cliffs, Prentice Hall.

I. Dilli, H.-J. Hoffmann (1995): DIADES II: A Multi-Agent User Interface Design Approach with an Integrated Assessment Component. In: SIGCHI Bulletin (In print)

H.Dunckel, W. Volpert(1993): Konstrastive Aufgabenanalyse im Büro. B.G. Teubner, Stuttgart.

P.Gorny, A. Viereck(1987): Eine Vorgehensweise zur Entwicklung interaktiver Programme. In: K.-P. Faehnrich: Software-Ergonomie. State of the Art Nr. 5, Muenchen, R.Oldenbourg. 93-105.

P. Gorny, A. Viereck, L. Qin und U. Daldrup(1993): Slow and Principled Prototyping of Usage Surfaces: a Method for User Interface Engineering. In: Zuelighoven: Proceedings RE'93 - Prototyping. Bonn April 1993. Stuttgart, B.G. Teubner, 125-133

P. Gorny(1994): Principled Design of Ergonomic User Interfaces. In: B. Zajc, F. Solina (Eds.): Proc. ERK '94 - IEEE-Confenrece on Electrotechnical and Computer Science. Portoroz, 1994. Ljubjana, IEEE Slovenian Section. 27-34.

R. Iannella(1994): HyperSAM - A Practical User Interface Guidelines Management System, in Proc. of the 2nd Annual CHISIG (Queensland) Symposium, Bond University, Australia.

Ch. Jannsen, A. Weisbecker, J. Ziegler(1993): "Generating User Interfaces from Data Models and Dialogue Net Specifications. In: S. Ashlund et al (Eds.): Proc. INTERCHI '93. April 1993. Reading etc., Addison-Wesley. 418-423.

J. Long et al.(1992): Integrating human factors with the Jackson System Development method: an illustrated overview. Ergonomics. Vol. 35(1992), 1135-1161.

R. Oppermann(1992): Software-ergonomische Evaluation. Der Leitfaden EVADIS II. Berlin, NewYork, DeGruyter.

H. Reiterer(1995): IDA - A design environment for ergonomic user interfaces. (In this volume)

J. Vanderdonckt(1993): A Corpus on Selection Rules for Choosing Interaction Objects. FUNDP University of Namur, Inst. of Computer Science. TR 93/3.

J. Vanderdonckt(1995): Accessing Guidelines Information with SIERRA. (In this volume)

H. Wandke, J. Hüttner: Unterstützung bei der Gestaltung von Benutzungsschnittstellen. In: H.-D. Böcker (Ed.): Proc. Software-Ergonomie '95 - Darmstadt Februar 1995. Stuttgart 1995, B.G. Teubner.