COOPERATIVE PROTOTYPING EXPERIMENTS

- USERS AND DESIGNERS ENVISION A DENTAL CASE RECORD SYSTEM

Susanne Bødker and Kaj Grønbæk

Department of Computer Science,
Aarhus University, Denmark*

ABSTRACT

This paper describes experiments with a design technique that we denote *cooperative prototyping*. The experiments consider design of a patient case record system for municipal dental clinics in which we used HyperCard™, an off-the-shelf programming environment for the Macintosh. In the experiments we tried to achieve a fluent work-like evaluation of prototypes where users envisioned future work with a computer tool, at the same time as we made on-line modifications of prototypes in cooperation with the users when breakdowns occur in their work-like evaluation.

The experiments showed that it was possible to make a number of direct manipulation changes of prototypes in cooperation with the users, in interplay with their fluent work-like evaluation of these. However, breakdowns occurred in the prototyping process when we reached the limits of the direct manipulation support for modification.

From these experiences we discuss problems in the process, requirements for design tools, and issues involved in getting going with cooperative prototyping with active user involvement.

1. Introduction

A while ago, one of us and some of our colleagues gave a talk to some information system designers about system design with users. The talk contained a small role play where an office worker, and a systems designer made on-line modifications of a prototype in between the office worker's test of the prototype. This dialogue caused much amusement in the audience, to our surprise not because the systems designer in the play messed up the situation at the end, but because the systems designers in the audience found it too far out to try to modify prototypes in cooperation with users.

Challenged by this experience we will describe some experiments which demonstrates that it is possible for users to envision a new computer application and its use at an early stage in system design, and that it is possible to do on-line modifications, even in a prototyping process where cheap 'off-the-shelf' software is applied. Moreover, we will discuss the problems that we encountered in the process, and how this design approach raised new requirements on design tools and techniques.

The experiments took place at two trade union courses and the primary purpose was to let dental assistants experience the possible use of computers in their work. Thus the primary purpose of our experiment was not to let the users participate in a real systems design process, and we do not here describe a full systems design process. We only discuss aspects of the process which aim to illustrate the potential use of a cooperative prototyping approach early in a system design process. In a real design process, this prototyping should follow a process of mutual learning, and be applied together with other tools(Bødker et al.,1987 and Kyng, 1988).

Before we continue with the description and discussion a short motivation for this approach to system design with users is given.

1.1. Why prototyping with users ?

We are currently working in a research program called "Computer Support in Cooperative Design and Communication" (Bøgh Andersen et al., 1987, Bødker et al., 1988). One of the aims of this program is to

* Ny Munkegade 116, DK-8000 Aarhus C, Denmark. Phone: +45 86 12 71 88. Email: bodker@daimi.dk, and kgroenbaek@daimi.dk.

develop tools and techniques to support systems design as a cooperative activity between users and designers, the purpose of this being to improve the quality of the computer applications, and thus of the future use situation.

We view design and thus prototyping as a process in which designers and users in cooperation determine and create the conditions for the future work in an organization. The computer system that is designed is part of these conditions, but also education and organization of work deserves to be mentioned. To find out how a future computer system improve the work of the users, the users must be able to somehow experience this, i.e. to envision the future use situation. To envision a future use situation is *neither* to read a description of a computer system or of its use, *nor* is it to see a demonstration of a prototype: Human beings possess skills that are not easily articulated or imagined without actually being in a situation of acting. For example, humans normally use tools in the work process without being aware of them, skills of using the tools are usually not articulated, they are triggered in the work situation when dealing with a certain piece of material. To make such skills contribute to the design of a computer system we have to try out the skills by actually acting in a situation. Thus good design means that it is important to set up situations which make it possible for the users to gain hands-on experience with mock-ups or prototypes, modeling the future computer application early in a development process (Bødker, 1987 a and b, Ehn, 1988). While being "in the future" conducting a future work task by means of a prototype certain breakdowns (Winograd and Flores, 1986) will occur, by which the fluent conduction of the work task will stop. As an example we may become consciously aware of the tool that we are using because it does not fulfill its purpose. In a prototyping process such a breakdown may lead to a change of the prototype, and eventually to a change of the overall design of the future computer system.

This view of prototyping goes much further than prototyping approaches seen in current practice of system design. Examples showing this is found in an empirical research project on the use of fourth generation systems for prototyping (Grønbæk 1988). In the projects studied, prototypes were almost exclusively developed by the designers *on their own* and the users were mostly confronted with these prototypes in *demonstrations* at project meetings. Thus the users were only involved in a passive evaluation of prototypes which often lead to a silent acceptance of the prototypes as basis for implementation. In many of these cases users raised new requirements to the systems when serious breakdowns occurred in the use of the system during or following the installation activities. We believe that many of such breakdowns could have been anticipated if the users had been actively involved in the prototyping activities in cooperation with the designers early in the design process.

1.2. Cooperative prototyping

The aim of the technique which we denote cooperative prototyping is to establish a design process where users are participating *actively* and *creatively* in cooperation with designers. The idea of the approach is to rapidly develop one or more mock-ups or prototypes, modelling the future system, that can immediately be related to core tasks of the use domain. The idea is to establish a process with a work-like evaluation of these mock-ups or prototypes. Breakdowns, caused by a bad or lacking design, are turned into rapid modifications of the prototypes. This process continues until the group decides that they has a sufficiently clear idea of what the system should be like, and the designers have an idea of how it can be implemented.

This kind of process is conducted by a group of designers in cooperation with a group of users, and the process is characterized by a close coupling between design and evaluation activities. Cooperative prototyping requires that the group of designers and users have access to tools that support rapid development and modification of prototypes. The prototypes should provide some minimum of (simulated) functionality that makes it possible to envision future work tasks. This requirement determines a need for developing computer based tools for the process. Modifications should either be made on-line in combined evaluation/design sessions if appropriate, or immediately after the session enabling a new evaluation "the day after".

Cooperative prototyping is meant to combine the ideas of using computer-based tools for exploratory prototyping (Floyd 1984) with approaches to design that allow users to participate in the modification of wood and paper mock-ups as described in (Bødker et al., 1987). There is only few descriptions in literature of examples on design groups both using computer-based tools for exploratory prototyping and having the users participating actively in the sessions where evaluations and modifications are made in a close interplay. In our research program one such example is documented in (Hauerslev and Jacobsen, 1988). Here prototyping experiments on the design of a telephone switchboard for a secretary were performed in close cooperation with a user, utilizing a domain specific programming tool built on top of a programmers interface to a network supported graphics package called NeWS running on SUN workstations. In this example a user was evaluating a prototype running on a graphical workstation, and modifications needed due to occurred breakdowns were made on-line by a programmer programming on another connected workstation in the same room.

In our research program we want to develop better computer support and techniques for this cooperative kind of design with users. The initial steps in this research and design process are to explore some of the difficulties and benefits of doing design with users utilizing existing computer-based tools. The experiments described in this paper are another example of such an exploration where we in contrast to the example described above use a direct manipulation supporting tool. The tool is HyperCard™ which is a flexible and cheap tool available for the Macintosh.

2. The case study

In order to help the reader understand the prototyping experiments we give a brief description of the application domain of our studies, the municipal dental clinics. Furthermore we shall present the setting of our case study. To illustrate the contributions of the users and the modifications made to the prototype during and in between the sessions, a more detailed description of scenes from our prototyping experiments, is given afterwards.

2.1. Municipal dental clinics

In the experiments we were developing a prototype of a patient record system for municipal dental clinics. The users are mainly dental assistants working in dental clinics of public schools. The work of the individual dental assistants vary a lot depending on the size of the school, and the organization of work at the clinic. All the dental assistants in these clinics were women, most of them with no real previous experience with computers.

In the dental clinics and their municipal administration two computer systems are currently applied: All municipal dental clinics are using a centralized statistical system based on off-line registration of tooth treatment on OCR-forms[1] representing the teeth of the patients. The dental assistants fill in the OCR-form after a patient has been treated. The forms are sent to a private company which is responsible for automatic reading of the forms and computation of the statistics. Another centralized system has just been developed and is used in very few clinics. This system contains a kind of patient record with basic information on patients such as name, address, parents, school, grade, movements between schools/clinics, etc. In the clinics which use this system the dental assistants do not have on-line access to the patient information at the moment, instead the system is operated by a secretary on request from the dental assistants. These two systems run totally separate and have no interaction at all.

The dental assistants have quite a few reasons to be critical against these systems: None of the systems serve as proper tools in the daily work of dentists or dental assistants, the patient case records still have to be kept on paper, and no dental assistants have been involved in the design of the systems although dental assistants constitute the main group of potential users.

In fact many dental assistants want computer support for their work, in particular for trivial administrative tasks so as to get more time for prophylaxis, treatment, and other patient contact. However, the dental assistants as a group do not have resources to design computer support for themselves. Trade union activities are their only possibility to influence design. Our experiments took place as parts of two consecutive trade union courses which aim to improve the dental assistants range of possible influence on technology development and use in their work. For descriptions of the background of such courses, see (Kyng and Mathiassen 1982, Ehn and Kyng, 1987, Kyng, 1988).

The design experiments are primarily part of a learning process where the dental assistants experience aspects of their future work with computer support, as well as they experience actual participation in design. The experiments deal with a decentralized patient record system, combining administrative information with treatment-oriented information. This type of system falls somewhere in between the traditional administrative records, which helps keeping track of the patients, the data collection for statistics, and the case record where also medical information is kept. For accounts on similar systems for health center physicians, see e.g. (Bajlum and Nielsen 1988) or (Engeström et al. 1988).

2.2. Setting of the prototyping experiments

We have been teaching a number of union courses to dental assistants without using running computer examples. Partly motivated by our concurrent research program we decided that future courses should illustrate both concrete examples of alternative computer technology and examples of design processes with user involvement. For this purpose a prototype of a decentralized patient record system running on a Macintosh computer was developed. HyperCard™ was the basic tool used for the design of the prototype, but also Reports™, a scanner, and a paint program for the Macintosh were used[2].

A good reason to choose HyperCard is that its card metaphor fits well in with the metaphor that we normally use, when teaching the dental assistants how a computer database works: The computer memory is pictured as a

[1]OCR=Optical Character Recognition.

[2]See (Goodman 1987) for a description of HyperCard™ and (Reports 1988) for a description of Reports™

paper file box containing cards which contains named fields. From the file box cards can be read, written on, added to removed from, etc. We teach "programming" by means of structured Danish pseudo-code which allows us to access the file boxes, cards, and fields. Moreover, the dental assistants are familiar with card and file boxes in their work.

In our description of the experiments we will, for the first course, focus on data entry and on-line retrieval facilities and for the second course on reporting facilities. This partly reflects the emphasis in the experiments as well.

2.3. Prototyping of data entry and on-line retrieval parts

In this and the following section we describe the experiments that took place at each of the two courses. We have chosen to write the story as if it was one of us who went through the whole process. In the description, the 'I' person is the computer scientist [3]. Furthermore we have chosen to put in some dialogues which are, although not strictly authentic, meant to illustrate what happened. The documentation of the experiments consists of our hand written notes and the different versions of prototypes, only. The ideal documentation of the experiments would have been video recording, but this kind of recording would have disrupted the teaching aspects because for the dental assistants attending the courses, this would have been yet another disturbing factor.

Preparation of 1st prototype

In order to prepare the experiment for the first course I did the following:

- reused a set of HyperCard stacks, made in a students project on the design of a prototype case record system for general practitioners (Bajlum and Nielsen 1988)
- scanned pictures of the upper and lower parts of a human set of teeth from an anatomic atlas into MacPaint documents.
- modified the MacPaint documents to have each of the mouth parts fit onto a card in HyperCard.

This 1st version of the prototype consisted of five different stacks and six different card types:

(1) A stack with one card as the start up interface to the system. This card contained a button to search patients with the name as a key, and a button to generate a new record entry for a patient with the appropriate set of interlinked cards. See Fig. 1 of appendix A[4].
(2) A stack with a card type consisting of fields for basic patient information such as name, address etc. See Fig. 2 of appendix A.
(3) A stack with a card type containing the picture of the upper part of a human mouth. On this each tooth was linked to a card of a type containing a description of the treatment history for the tooth. See Fig. 3 and 4 of appendix A.
(4) A stack with cards containing information on the lower parts of a human mouth similar to the previous stack. See Fig. 5 of appendix A.
(5) A stack containing cards with fields to register plans for future treatment sessions. See Fig. 6 of appendix A.

When the button to create a new patient record (see Fig. 1 of appendix A) is pushed, a set of four main cards from the stacks 2)-5) is automatically interlinked with buttons to jump between the cards for this single patient. Moreover, a set of three cards in each of the stacks 3) and 4) is allocated to show the treatment history for three selected teeth in each of the upper and the lower parts of a patient's mouth. Note that the idea was to show the principles of direct representation of teeth on the screen. The limitation to three teeth is an arbitrary choice. I entered a small set of fictious test data into this 1st version and brought it to the first week course. Altogether these preparations required less than two days of work.

At the course a short introduction to computer technology was given using examples from the dental clinics. Moreover, strengths and limitations of the two existing systems (see section 2.1) were discussed. At the second day of the course I gave a short demonstration of the prototype to the whole group of 17 dental assistants. Then I encouraged the dental assistants to come and work with the prototype in groups of 2-4 to get a better feel for the prototype and the technology in general. Some of the groups just tried to enter data into the system and search through existing data. I tried to stay on distance in order to let the dental assistants themselves explore the prototype. A few of the groups worked quite enthusiastically with the prototype and came up with constructive suggestions that I build into the prototype on-line or after the sessions.

---

[3]In section 2.3 "I" denotes Kaj Grønbæk who was the teacher of the first course, and in section 2.4 "I" denotes Susanne Bødker who taught the second.

[4]The figures are made from our prototype as it was right after the second course. Thus the text of buttons, fields, etc. is kept in Danish, and the set of fields and buttons in the stacks are little different from the initial prototype described here.

Examples of on-line modifications of the prototype

In the following I will describe scenes from the prototyping sessions to illustrate examples of the kind of changes that were made on-line together with the dental assistants:

• Change of the teeth representation: At some point two dental assistants were sitting together in front of the screen and I stood just behind. They switched between the cards containing the teeth pictures. Suddenly one of them said: "*The lower mouth part is turned upside down - that's quite confusing according to how we number the individual teeth !*"[5] I had turned both of the upper and lower jaws the same way with the front teeth pointing upwards. This was not satisfying for the dental assistants. One of the dental assistants explained: "*I want to think of the pictures of the teeth as I'm looking into the mouth of the patient when I look at the screen !*" I replied: "*Hold on for a moment ! - let me have the mouse*" then I entered the background of stack 4) containing the scanned picture, selected the whole picture, applied an operation called FLIP VERTICAL followed by application of a FLIP HORISONTAL operation, making the picture turn right. The dental assistants said: "*Wow !*" I dragged the transparent buttons linking the teeth to their history card to the right positions. One of the dental assistants said: "*Amazing - but what about the other patients - do we have to change the pictures for them, too ?*" I replied: "*No have a look at this other patient - the modification affected all patients !*" The whole modification activity lasted only a few minutes and I did not even touch the keyboard. One of the dental assistants took over the mouse again to continue playing with the prototype.

• Exploring alternative representations of tooth treatment: One of the dental assistants asked at some point while she and two others were sitting in front of the screen: "*Is it really necessary to enter a new page to describe the treatment for a tooth ?- it would be easier just to mark it directly on the particular tooth !*" I replied: "*Ehh No ! We can try to see how it would work out if we use the freehand drawing facility, but... - Can I have the mouse for a second ?!*" I entered the HyperCard freehand drawing tool and demonstrated how to make a direct marking of e.g. a tooth-filling on a tooth and how to get rid of it again with the eraser. One of the dental assistants got hold of the mouse again and tried to draw a big black spot on top of a tooth while she said: "*This is fun - have a look at this tooth it's completely rotten....Ha Ha*".

The example lead to a discussion between the dental assistants of advantages/disadvantages of either of the two representations of tooth treatment. The need for a more structured representation of each tooth to be able to be more precise in the marking was also discussed. I explained that for a real implementation of the direct marking idea it would be necessary to partition each tooth into its anatomic areas and have a data-structure mirroring the graphical marking in order to be able to generate the statistics. The current prototype was of course quite primitive regarding the possibility of exploring the direct marking idea, but the fact that both the button and the direct marking facility could be illustrated, started a valuable design discussion a concrete basis where the dental assistants were pointing at the prototype referring directly to the prototype. We actually concluded from this discussion, that a combination of direct marking and underlying linked descriptions would be a good solution, but one card per tooth was far too much space to allocate.

• Adding new data fields: Quite a few of the dental assistants, trying out the prototype, came up with suggestions of the form: "*We need to have a field for......!*" In these cases I asked to get control of the mouse and responded: "*Where do you want to position the field on the card?*" With the field tool of HyperCard I placed the required field on the cards and then asked: "*What text should we associate to this field?*" After entering the new field and text the dental assistants continued playing with the prototype. Fields such as: Doctors name/address, ID number of school, ID number of Local Government, and regular interval between dental visits were added like this.

• Changing layout of cards: Some of the dental assistants asked me to move fields that they found improperly positioned. We made these moves in the same way as we had added fields.

• Modifying button scripts: At some point a dental assistant was entering a new patient into the system and suddenly she said: "*Uh! My new patient disappeared???*" I walked up to her and asked: "*What did you do?*" The dental assistant: "*I'm not sure....but I think I pushed the Delete button instead of the Upper Mouth button !*" It then appeared to me that the delete patient button did not have a parachute accident, and I asked: "*I think I can make a fix so the program will ask you if you really mean the delete operation, but I need to be in charge of the keyboard to do that!*" The dental assistant said: "*Ok!*" I then started to edit the delete button, programming a parachute which is three lines of HyperTalk code[6]. The dental assistant was sitting beside me. This programming was done rapidly enough to keep her patience and when I had made the fix I asked her: "*Try to delete one of the other patients now!*" She then entered the main card of another patient and pushed the delete button and said: "*If I*

[5] All the citations are constructed by the authors from notes and memory.

[6] HyperTalk is the scripting programming language of HyperCard

push CANCEL now the patient wont be deleted... right ?" "Right !" I responded and she continued exploring the prototype.

• Copying buttons: At some point a group of dental assistants asked: "*Can't we search a patient on the 'Social Security Number' instead of the name - we often need to do that !?*" I responded: "*For the moment you can't, but I can probably easily make this extension for you - let me have the mouse !*" Then I made a copy of the 'Search on Name' button of stack (1), started the editor on the button script. I selected the label text of the script and asked the dental assistant who was in charge of the keyboard to type the new label text 'Search on Social Security number' and I repeated this for the name of the target field for the search. I then closed the editor and said: "*Try to push the new button now !*" The dental assistant pushed the button: "*Amazing - it asks for a Social Security Number'- do any of you remember the Social Security Number of Peter that we just entered in the system ?*' The exploration continues.

....A little later a person from the same group of dental assistants asks: "*Do we really have to return to the front-page*[7] *to search a new patient???*" I responded: "*Ehhh.. I'm afraid so.... for the moment that is the only way!*" The dental assistants: "*That's annoying ! Can't you just make one of these buttons to jump to the front-page!?*" I said: "*I see your idea..... - but I think what you really want is to have an item on a menu like this* (pointing at the menu bar) *so that the search facilities are available independently of which card you are on! But the menu solution is to hard for me to make right now- so maybe we should make the button solution temporally*" I then copied a button from stack (2) to jump to the card of stack (1) to the backgrounds of all the other stacks to make easy access to the search facilities which in this early version were available only on the "front-page" of the record system.

Post session modifications of prototype

A number of suggestions and ideas that came up during the sessions could not be integrated in the prototypes on-line. The following are examples of this:

• I discovered a problem with some of the field modifications that were made on-line. Not all fields in the 1st version of the prototype were located in a background of a stack. This implied that a movement or change of size did not affect all the cards sharing the same background. Moreover, I had to throw away some test data, when I moved a field from the card level to the background level. I decided to clean these failed modifications after the sessions with the dental assistants, because it would be too confusing and time consuming to do on-line.

• The implementation of a menu with items to search for patients independently of which card the user was on was made after the sessions because it would be to time consuming to program global menus on-line. Moreover, it could not be done with direct manipulation as the majority of the changes were made on-line.

• During the sessions some dental assistants asked how to get hardcopies of certain combinations of the data stored in the system. I had to tell them that I did not bring a printer, and that the reporting facilities of HyperCard were quite primitive. Thus we could not experiment on these aspects of the system at the moment. Such experiments were instead prepared for the next course utilizing the Reports™ program for HyperCard.

Summary of experiences

In this first experiment we saw that with a small amount of preparation, access to cheap 'off-the-shelf' software, and borrowed pieces from other applications it is possible to create a situation where users and designers in cooperation can envision a future work situation. Breakdowns occurred at two levels: the first being where the unreflected use was interrupted, and the users asked for a change in the prototype. Some of these lead to a second kind of breakdowns, when the users lost interest in the change/design situation. Such breakdowns happened when new aspects had to be programmed in HyperTalk. Some of the first kind of breakdowns were on the other hand, handled rather well by the users and designers in cooperation: adding and moving fields, buttons and texts - changes that could be done with HyperCard's point and select facilities.

2.4. Prototyping experiments on reporting facilities

In this section we describe some of the experiments carried out at the second course. In this course a similar procedure as in the previous course was followed. What we will describe here is the changes in relation to reporting.

[7]The card of Stack (1), see fig 1 of Appendix A.

Preparation of the 2nd prototype with Reports™

The 2nd version was supplied with report facilities, utilizing Reports™ that allows us to make reports based on HyperCard stacks, by simply outlining a heading section, a details section iterating over the data fields specified, and a footing section.

The outlining is done with a WYSIWYG drawing/writing tool which also allows for laying out fields where information from HyperCard's fields can be filled in (this is done by pointing and selecting the name of a HyperCard fields from a given background). There is also a way of sorting or selecting according to certain fields. All is done by point-and-select (See example in appendix B).

Before leaving to teach the second course, I prepared by doing some necessary changes to the HyperCard stacks: First of all, the fields that I wanted to print out had to be named background fields, which they were not. Quite cumbersome, indeed. To prepare myself, I made a few simple reports to present to the dental assistants: a folder label, which would probably be useful for the dental assistants and a list of all the "Hansen"s to demonstrate how selection worked. The latter, I knew, would probably be of no use to them.

Examples of on-line modifications of the 2nd prototype

At the course, I started out by demonstrating the prototype to small groups of dental assistants. The dental assistants used the prototype and entered a number of data sets into the prototype. Some changes were made on the cards of stack (2). They were all rather small: one group added a new field for day care address, another information about bussing and a third group wanted to add information for orthodontics. In this process, the card became pretty crowded, and we had some discussions about what to do with that. One of the women suggested to put much of this information on a separate card, and I had to admit that that was too hard to do on-line. Our primary focus was on two kinds of reporting activities:

•Layout modifications: The women in groups tried out my reports and we made a variety of reports. We started out with some simple things such as address labels for their current manual case record folders. Much of the time was spent experimenting with what information was needed for that labels, and how this should be laid out - how big should a label be, which information should be put where, etc. Even though I was present the dental assistants were capable of laying out the fields themselves.

•Query formulation and modification: The dental assistants wanted to have lists sorted according to birthday, lists of people who have to be called on within 3 months from date, and lists of 5th graders, who are called for for an extra orthodontics check. By then the 'programming' started to be too complicated for the dental assistants and I had to take charge of the process: *"What does it mean to be called on 3 months from now?" "Well, now is November 16 1988, so we need to find the children who are marked with a called on date before February 16 1989..."* Somebody answered. *"Yes"*, I said *"but the only field we have is one saying how many months remains before the next call" "But then we need to write out those where this number is smaller than 3"* the woman answered after a while. With the lists of 16 year olds, who are to start a different way of seeing the dentist, etc. it became even more complicated, because we needed to extract the age from the social security number, masking out certain digits. Furthermore, the direct manipulation way of doing these expressions allows only for very simple expressions (AND and OR of conditions with no parentheses). More complicated expressions can be made in a special scripting mode, where one writes HyperTalk code. For various reasons I did not try this in the course, partly because I expected it to be too complicated for the dental assistants to follow, and partly because the debugging facilities for these scripts are very bad.

Generally the dental assistants were very interested in designing these reports, but programming of conditions and sorting were too hard for them to do on their own, partly because of Reports and partly because of lack of familiarity with logical expressions.

Up until that point we had only made reports with data from one type of cards (one stack). And of course one group realized that they wanted to print out a survey of the treatment of all the teeth of a patient. I used much effort to tell them that this could and ought to be done, and that theoretically it is possible to do direct manipulation reports with fields from several stacks, but that in our version this keeps creating a systems error. Scripting reports is possible, but again I didn't feel like doing that in front of the dental assistants. Yet another group suggested that we made lists of who had a certain treatment made to a certain tooth. Again my conclusion was: *"Yes, in principle it can be done, but I bet you don't want to wait for this"*. For this reason many good suggestions for reports were never realized in the prototype.

Summary of experiences

In this second experiment the designer played a slightly different role than in the first experiment. Primarily this was because the experiments here were focussing on how to use a prototyping tool to build new aspects of the prototype, and not about how to use and change the prototype. Again the users were fully capable of experimenting with the direct manipulation design of reports. Again, when breakdowns occurred it was either because point-and-select did not suffice, or it was in the transfer between HyperCard and Reports, which are unfortunately not designed to be fully integrated.

3. Successful aspects of the experiments

Cooperative prototyping requires a different commitment and involvement from the users than traditional interviewing/description techniques: On the one hand the users should be motivated to play the game of being in a work situation with a preliminary prototype of a future computer application, on the other hand it is necessary that the users know the prototype, they are playing with, is changeable and that it is far from being a complete application. We will briefly discuss our experiences on how to get this kind of process going.

3.1. Getting a fluent work-like situation going

Before the experiments we expected that the major challenge would be to keep the unreflected daily action of the users going in a prototyping process when, at the same time, we often have to stop and make changes to the artifact they are using. This problem is described in (Hauerslev and Jacobsen, 1988).

In our situation we could not go out into the real setting and have dental assistants use the prototypes there, and also the environment was not suitable for simulating a real dental clinic. It was the fact that the dental assistants were together at the course that made it possible for them to 'be in the situation' by having to demonstrate to each other how they did things 'at home'. Thus the setting of the experiment is having the characteristics of a *laboratory experiment with modifications* according to (Bødker et al.,1988, Grønbæk, 1988). Clearly the situation could be more ideal in this respect. But it is even more important to get the prototype working so that the dental assistant's being in the situation doesn't break down too often due to problems with the prototype that would not be problems with a real application. Our preparation seemed to be reasonably successful in this respect, and it was possible for the dental assistants to step into the illusion that they could perform realistic work tasks with the prototype. This is stressed with the fact that we during both courses got questions from the dental assistants such as: "*Where are they using this system?*" and "*What 's the price of the system?*" although we had told them that it was a preliminary prototype. The real challenge was to interrupt the process and modify the prototype, when they were criticizing some aspect of it, and then start the evaluation process again.

By letting the dental assistants sit down and try out the prepared prototype, while doing something which resembles their daily work tasks, they could hands-on experience their future, e.g. with what to look for when they open the case record folder for a specific child. These tacit knowledge contributions to design is discussed by a number of authors (Bødker 1987, a and b, Ehn, 1988, Winograd and Flores, 1986, Polanyi, 1967). Using a prototype providing familiar cards and pictures of teeth made it possible for them to formulate quite specific needs and requirements to a future computer application by using their own language and pointing at the prototype.

3.2. Doing on-line modifications

We expected that using HyperCard and Reports would make it possible in a number of situations to discuss and design solutions to the dental assistants problems with the prototype right away.

Our main concern regarding this was that it would be difficult to keep the patience of the dental assistants during the modification activities, i.e. it was our expectation that it would be hard for us to do major programming in the sessions with the dental assistants[8]. The level of preparation, however, allowed us to do many of the needed changes through direct manipulation and without loosing the patience of the dental assistants. As expected it caused some problems when major HyperTalk programming was needed, and also in a case where one of us got lost in trying to create a new card type for patients and have it linked up properly for the creation procedure. Experience number one is *that the designer must know the prototype and the prototyping environment well enough to recognize the complexity of a change and not start programming activities that destroy the users' understanding of the situation.* Moreover, with the present tools, it was clear that a certain clean-up of the prototype versions were needed. The changes made when the dental assistants were present were not always as nice as they could be. Often we chose a simple solution for the moment aiming to improve it later. At times, even ma-

[8]This problem is described in (Hauerslev and Jacobsen, 1988).

jor structural changes were needed. This was quite complicated with HyperCard, without loosing the test data that the dental assistants had entered. In other words, with this type of prototyping there is still quite a lot of work for the computer persons in between meetings.

We see *direct manipulation facilities* as provided with HyperCard/Reports as quite important for a cooperative prototyping process with on-line modifications[9]. Our primary source of comparison used a two workstation situation where the designer modified the prototype by programming on one workstation, and the user evaluated the prototype on the other workstation in parallel (Hauerslev and Jacobsen, 1988). Compared to this, we did not need to hide the process of prototype modification to the users, because the direct manipulation facilities made it possible for the users to follow the process as most changes can be made fairly quick this way. The direct representation of the data structure onto the screen (the cards, fields, etc.) was valuable in this case. We can easily see situations where the card structure would be a limitation, but in this case the direct mapping of 'good old-fashioned' cards on the screen made it relatively easy for the dental assistants to understand what the prototype could do for them and how the direct manipulation changes affected it.

3.3. Outcome for the dental assistants

To illustrate the outcome of the experiments with regards to the dental assistants, we will mention some tentative conclusions on the overall idea of the proposed kind of computer support for the work of dental assistants. These are issues that we discussed with the dental assistants during and after the experiments:

- The user interface idea with cards and direct representation of the teeth in a computerized case record for dental clinics seems quite promising, but a full-scale system needs to be able to communicate with centralized databases in order to, e.g. compute the required statistics, and keep track of children moving between clinics/schools. Actually, the activity of keeping track of these movements requires much administrative effort from the dental assistants at present.

- The designer's idea of having each tooth on a separate card did not fulfil the requirements of the dental assistants. The dental assistants needed to be able to get a quick overview of all the treatment that has been given to a patient. This kind of overview could be provided with direct marking on the teeth pictures, but it would require a more structured representation of the teeth, i.e. the anatomic structure of each tooth should be mirrored precisely in the pictures and underlying data-structures. However, it would still be necessary to combine this direct marking with tooth buttons to jump to a field for more detailed information described verbally on an underlying sample card representing all the teeth of the current patient. A major restructuring of the prototype seemed to be necessary to fulfil this demand from the dental assistants.

- The basic patient information in Stack (2) need to be partitioned between two different card types in order to preserve the general view of the information. The suggestion from some dental assistants was to move address information (i.e. parents and general practitioners) to a secondary card that one can access with buttons similar to the teeth picture cards.

Through the discussions that took place when problems was encountered, both the dental assistants and we got to know more about how, specifically, such computer applications should be shaped to fit the specific work situation. Another equally valuable outcome from the design experiments was that the dental assistants improved their own personal resources to be able to influence technology issues at their own work place and as a group of professionals. They got on the one hand an impression of potential alternatives to the earlier mentioned systems known for their domain (see section 2.1), and on the other hand a feel for how to participate in and influence system design.

3.4. A design process in general

A prototype used the way we have described it becomes a valuable vehicle of communication and stimulation of user enthusiasm. Through the communication process the dental assistants learn about the different ways a certain type of computer application can be used in their daily work. We claim that the above conclusions together with the last version of the prototype would be a valuable input for a design process aimed at implementing a good tool for the daily work of the dental assistants.

The cost of reaching the outcome and building the prototypes was low compared to the benefits, given that we had some knowledge of the application domain in advance, as well as access to a prototype for a similar application domain. We spend approximately one week full time work on the design activities and experiments. In a case where the designers do not have access to this kind of knowledge on beforehand, it is of course necessary with a learning process to understand the domain. It is our strong belief that we did not at all utilize the full po-

[9] See (Schneidermann, 1983) or (Hutchins et al., 1986) for a discussion of the direct manipulation concept.

tential of design ideas from the dental assistants during these two limited experiments. We could still make several iterations on the user interface design of the current prototypes to uncover more aspects of the dental assistants' needs for computer applications for their daily work. But unfortunately we only had a quite limited time together with the dental assistants at the two courses.

4. Potential breakdowns of cooperative prototyping

Although we describe the experiments as relatively successful there is a number of sources of potentially undesired breakdowns in cooperative prototyping, and there is a number of problems to overcome to integrate such prototyping approaches in system design projects in general. In the following we discuss some of these issues.

4.1. Reaching the limit of direct manipulation

From our experiences it seems that it is when we reach the limits of the direct manipulation possibilities that we get undesirable breakdowns in the design situation. Even though HyperCard uses a card metaphor as the basis, the users easily get lost in the process when the computer professional is starting to write code to manipulate the cards. And if this kind of coding takes more than a few minutes, the users will loose their patience. This leads to the idea that *direct manipulation should be exploited as much as possible in tools for cooperative prototyping.*

Rapid and direct manipulation of the prototype is crucial for cooperative prototyping, but it is not sufficient. In order to keep a process going where the users envision a future work situation it is crucial, too, to preserve the test data and examples related to their work when modifications to the prototype are made. Because of the quite close coupling between user interface and data structure in HyperCard we in some situations had the problem of not being able to convert (and even of loosing) parts of our test data during modification of the data entry and retrieval facilities. This leads to the conclusion that *test data should be stored in a separate database where the individual elements can be linked to the required user interface parts by direct manipulation.*

4.2. Interrupting the fluent work-like evaluation

To perform cooperative prototyping where modifications of the prototype are made frequently raises certain process problems. When the users work-like evaluation is interrupted there is a great risk that their illusion of being in the situation breaks or they loose the patience. Thus, attention should be paid to both finding proper breakpoints where the modifications can be added to the prototype, and starting only realistic modifications in order to keep the designer's disruption of the users' evaluation to a minimum. Related to this it is worth pointing out that the designer should be able to assess the amount of work involved in doing a certain change. Thus the designer must have a good understanding of the tools, by which he can assess the required changes. As an example from the case we have described it was important to know when test data was preserved or lost. To some extent we can say that the designer must know the design rationale behind the tools, and that a requirement to a good tool is that it provides appropriate and consistent ways for the designer to understand what is going on in the design situation.

Moreover, there is also the danger that the designer's attention will be fully directed towards the modification of the prototype. This is in particular the case if the tools allows for preparation of modifications in parallel with the users' ongoing evaluation (Hauerslev and Jakobsen, 1988). Here examples show that the designer's attention at some points was fully directed towards programming. All in all it seem to be a good idea to have a co-designer to facilitate the evaluation process as proposed in (Hauerslev and Jakobsen, 1988), when prototype modifications take place on-line.

4.3. Unrealistic expectations

The limited prototype that was developed in our experiments is not suitable to be part of a future application for a number of reasons, e.g. its lack of integration with real database facilities. An obvious disadvantage is that the prototype has to be thrown away when the experiences gained from it have been used in later prototypes which in turn are used for implementation of the final application. This makes it important for cooperative prototyping that the designers are able to adjust the *users' horizon of expectations* regarding early prototypes. An example mentioned earlier indicates that some dental assistants thought the current prototype was a product already available for purchase. If this kind of misunderstanding is not adjusted immediately it can be hard later on to explain to the user why it will take, e.g. two years to implement a full-scale system including database and networking.

It is important for the group to keep reminding itself that the current prototype is not the final application, and that it has limitations of different kinds which may or may not have relevance for the actual evaluation situation. The users should know, that just because the designer has prepared in a certain way, it does not mean that they have to accept the prototype as it stands.

Moreover, the exploratory prototyping approaches in general require that the computer professional has to have a good understanding of how realistic it would be to implement the features exposed by the prototype. This is to prevent unrealistic expectations (Ehn and Kyng, 1984, Bødker et al., 1987). Not because we should avoid experiments with features which are unrealistic or utopian, but because we need to discuss the trade-offs of such features (one conclusion in a certain case could be that implementing an application without this feature is simply not worth while).

4.4. Limitations of the design tools used

Although HyperCard and Reports are not the only tools to be used for cooperative prototyping we will briefly mention some of the specific problems that we had with these tools.

We discovered some problems with the *integration of the tools*. It has been mentioned that the card metaphor used by HyperCard seems like a good idea for this kind of application domain. However, the card structure in HyperCard raises problems in combination with Reports, because printouts often has to break with the card-in-stacks structure: how easy is it to e.g. write out all information about one patient ? Moreover, Reports function almost like HyperCard, but only almost, and the two are not fully integrated (and not fully separate). As an example it is not possible to copy HyperCard code between scripts in HyperCard and scripts in Reports. And Reports does even not support debugging the way HyperCard does it

There is quite a few *database issues* that cannot be explored with HyperCard as the prototyping tool. The most striking examples are multi-user access to the data in the system, and data security issues. These important data base aspects of case recording systems have to be envisioned with other tools. These observations advocates for the use of an integration of HyperCard with a 'real' database system[10].

The HyperCard tools does not provide *multi tasking* facilities, e.g. editing of more than one script at a time, and editing scripts concurrently with the execution of programs. Such facilities are needed to support preparation of prototype modifications in parallel with the users evaluation in line with the experiments described in (Hauerslev and Jacobsen 1988). We find such a parallelism useful and it can be utilized in situations where the users can proceed with their tasks using the prototype without interruption, while a change is prepared by the designer. In our experiments programming of button scripts for new functionality could have been prepared in parallel with the users evaluation, because the prototype was quite stable and could be operated by the dental assistants on their own.

The required parallelism could be supported, e.g. by a network with two workstations and multi-user access to the code of the prototype. These are facilities which are standard in more advanced programming environments such as Interlisp[11] and Unix[12]. These limitations of HyperCard make it harder to try to do even simple programming in sessions where users participate. The (only) realistic kind of programming, that we can imagine being possible in situations with users participating, is reuse of code fragments or copying the idea from one piece of code to another.

5. Getting going with cooperative prototyping

In this section we will outline what we find are the most important issues for getting going with cooperative prototyping experiments in the early phases of system design projects.

5.1. Preparation of experiments

The most important prerequisite to get going with cooperative prototyping is that a group of users and professional designers can work together without disturbance, e.g. that the users are free to participate. This was quite easy for us to achieve during these courses, thus our situation might be more ideal than system design practice.

Another prerequisite is preparation of the experiments. The designers must take the initiative to help the group as a whole get a mutual understanding of what the process is all about and to bring the users in a situation where they can envision a future work situation during the experiments. It was possible for us to our knowledge from earlier courses to prepare prototypes and test examples. In other cases it is necessary for the professional designers to spend much time with the users before getting to this point. In (Grønbæk, 1988) it is recommended that a working group consider and discuss at least questions related to the following issues when organizing experiments in system design projects: Purpose of experiment, extension of prototypes, selection of participants, preparation for participants, setting of the experiment, and evaluation criteria. In (Greenbaum and Kyng, forthcoming) a number of suggestions for how this early work of a working group can take form, are discussed.

---

[10] A integration of HyperCard with a relational database system called ORACLE (Trademark of ORACLE Corporation) that run on both mainframes and Macintoshes already exists.

[11] See (Teitelman and Masinter,1981)

[12] Unix is a trade mark of AT&T Bell Laboratories

Regarding the discussion of the extension of prototypes we will also point at the importance of exploring alternatives instead of freezing the first idea. In our experiments we did not develop alternatives in parallel, we only used the existing systems as sources of comparison. When needed, alternative prototypes should be developed and used to enhance imagination about different possible future use situations, which in turn helps the users to become active in the discussions.

The designers are the process experts who know how to focus the evaluation and discussions. Thus the designers should prepare themselves to play different roles such as conductor, designer, and programmer in the experiments such as discussed in (Hauerslev and Jacobsen, 1988). In our experiments the single designer played all the different roles, which might not be ideal when modifications are made on-line as discussed in section 4.2.

5.2. Access to domain specific objects

HyperCard can to some extent be viewed as a domain specific design tool for the type of application domain studied in this paper, because its basic structure suits well with the traditional structure of the artifacts of the domain. In general we would argue for domain specific objects in the design environment. If we consider the success of 4'th generation tools it can be explained exactly in this: they have been designed for large-scale information storage and retrieval applications, and standardized program fragments, related to the domain of information systems, are built in as primitives of the tools.

In the preparation of the 1st prototype we reused and modified a set of stacks already prepared for a related type of application namely a case record system for general practitioners(Bajlum and Nielsen, 1988). This reuse was a quite important step towards the rapid design of the 1st prototype. A useful facility for us in our preparation as well as in the prototyping sessions would have been a library of basic objects for the particular use domain combined with documentation and search/browsing facilities. We have partly build a collection of such objects in our preparation, and we would have needed to expand this further if we had gone on with the prototyping activities, e.g. scripting in Reports. The scripts for search criteria combining several stacks are complicated and hard to debug, which would make it almost impossible to write them from scratch in the prototyping situation.

5.3. Users in charge of the evaluation

In our experiments *the users have been in charge of the keyboard and mouse* during testing, except for initial demonstrations given to the groups. We find that there is a vast difference between the kind of user enthusiasm and response that we get in the case of demonstration and the case where the users are in charge of the prototype. Demonstrations are boring, but when getting their hands on the prototypes the users get a better chance of playing that they are in their daily situations, and thus, they get a better chance of using the skills that they have in the domain. The computer professional should act as a conductor and as an observer of the evaluation. When breakdowns occur for the users due to their need for more training he/she should intervene as a teacher. And when breakdowns occur for the users due to lacking or bad design he/she should intervene as designer or programmer.

The need to let the users get their hands on the prototypes requires that the prototype has a certain degree of robustness, at the same time, though, as the designer should be able to make the needed interventions. It is the professional designers who need the flexibility whereas it is the users that need the robustness in order to have any realism in their examinations.

6. Concluding remarks

We have described an example of a technique for system design called cooperative prototyping. Experiments with design of a case record system for dental clinics at municipal schools have been reported on. We have used the quite cheap off-the-shelf tools, HyperCard and Reports running on Macintosh workstations.

The tools allowed us to develop an initial prototype with an effort of two days of work based on our knowledge of the use domain. The initial prototype was on the one hand stable enough to be evaluated in a work-like situation, and on the other hand flexible enough to allow on-line modification. Thus we conducted a cooperative prototyping process where users used the initial prototype for work-like evaluation, and in cooperation with the designers we modified the prototype whenever breakdowns in the work-like situation occurred. This process created a remarkable user enthusiasm and response that we claim is valuable for the early stages of a design process in order to achieve a well-tailored system for the users. Thus we see our experiments as a promising example indicating that cooperative prototyping approaches may become realistic in the future of system design. However, it is not without problems to reach this goal and we need to go on improving our understanding of both cooperative design processes and the needs for tools to support them. We believe for the moment that the most promising direction to move is to go on building domain specific tools to try out the ideas in real design situations in cooperation with groups of people who work in these domains. Beyond these specific experiments we hope to be able to develop the ideas of cooperative prototyping, in line with the discussions in this paper.

Acknowledgements

We wish to thank Liam Bannon, Morten Kyng, Pelle Ehn and Kim H. Madsen for their valuable comments on earlier versions of this paper.

References

Bajlum, T. and T. Nielsen: *Edb-støtte til lægepraksis - et eksperiment med udvikling af en brugsmodel og en prototype*, University of Aarhus, 1988

Bødker, S. et al.: 'Computer Support for Cooperative Design', in *Proceedings from the Second CSCW*, (D. Tatar, ed., 1988)

Bødker, S. et al.: *A Utopian Experience*, in Bjerknes, G. et al. (eds.): *Computers and Democracy – a Scandinavian Challenge*, Gower 1987

Bødker, S.: *Prototyping revisited - design with users in a cooperative setting*, Report of the 10'th IRIS, Vaskievesi, Finland, 1987a (P. Järvinen, ed.)

Bødker, S.: *Through the Interface – a Human Activity Approach to User Interface Design*, University of Aarhus, 1987b, Lawrence Erlbaum, forthcoming

Bøgh Andersen, P. et al: *Research Programme on Computer Support in Cooperative Design and Communication*, University of Aarhus 1987

Ehn, P. and M. Kyng: 'The Collective Resource Approach to Systems Design' in Bjerknes, G. et al. (eds.): *Computers and Democracy – a Scandinavian Challenge*, Avebury 1987

Ehn, P. and M. Kyng: *A tool perspective on design of interactive computer for skilled workers*, in M. Sääksjärvi, ed.: *Proceedings from the Seventh Scandinavian Research Seminar on Systemeering*, Helsinki 1984

Ehn, P.: *Work-Oriented Design of Computer Artifacts*, Almqvist & Wiksell International, 1988

Engeström, Y. et al.: 'Computerized Medical Records, Production Pressure and Compartmentalization in the Work Activity of Health Center Physicians' in *Proceedings from the Second CSCW*, (D. Tatar, ed., 1988)

Floyd, C.: *A Systematic Look of Prototyping* in R. Budde et al., (eds.): *Approaches to Prototyping*, Springer Verlag 1984

Goodman, D.: *The Complete HyperCard Handbook*, Bantam Books 1987

Greenbaum, J. and M. Kyng (eds.): *Design by Doing*, Lawrence Erlbaum, forthcoming

Grønbæk, K.: *Prototyping with Fourth Generation Systems*, Daimi PB-270, University of Aarhus, 1988

Hutchins, E.L. et al.: 'Direct Manipulation Interfaces' in Norman, D. and Draper, S. (eds.): *User Centered System Design*, Lawrence Erlbaum, London 1986

Hauerslev, J. and N. Jacobsen: *SOKRATES: Sprogspil og Kvalifikationer*, University of Aarhus, 1988

Kyng, M. and L. Mathiassen: 'Systems Development and trade union activities', in Bjørn-Andersen, N. (ed.): *Information Society, for richer, for poorer*, North-Holland, Amsterdam 1982

Kyng, M.: 'Designing for a Dollar a Day', in *Proceedings from the Second CSCW*, (D. Tatar, ed., 1988)

Polanyi, M.: *Personal Knowledge*, Rutledge & Kegan Paul 1967

Reports - The Complete Report Generator for HyperCard, Activision Inc., Mountain View, California, 1988.

Schneidermann, B.: 'Direct manipulation: A step beyond programming languages' *IEEE Computer* August 1983.

Teitelman, W and L. Masinter: 'The Interlisp Programming Environment' in *IEEE Computer*, April 1981

Winograd, T. and C. F. Flores: *Understanding Computers and Cognition: A New Foundation for Design*, Ablex Publishing Comp. 1986

Appendix A

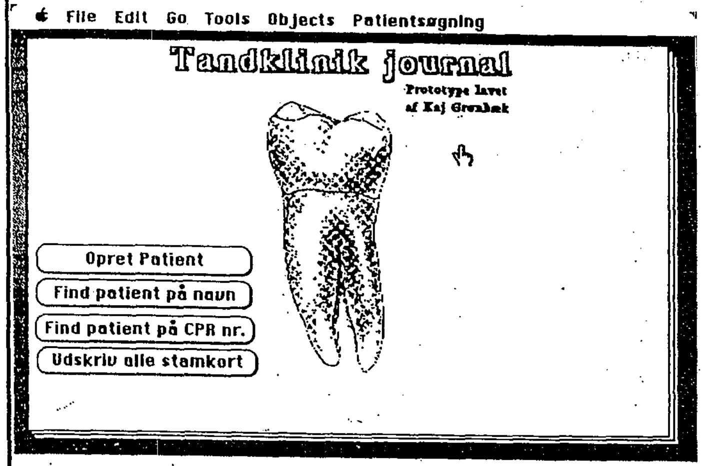


Fig. 1: Card of Stack(1)

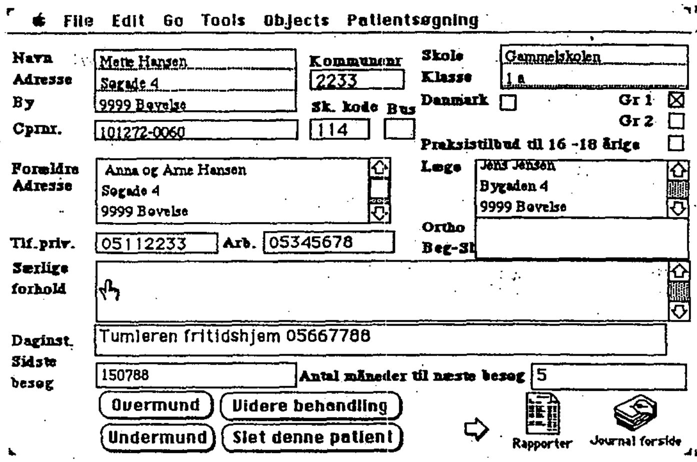


Fig. 2: Card of Stack(2)

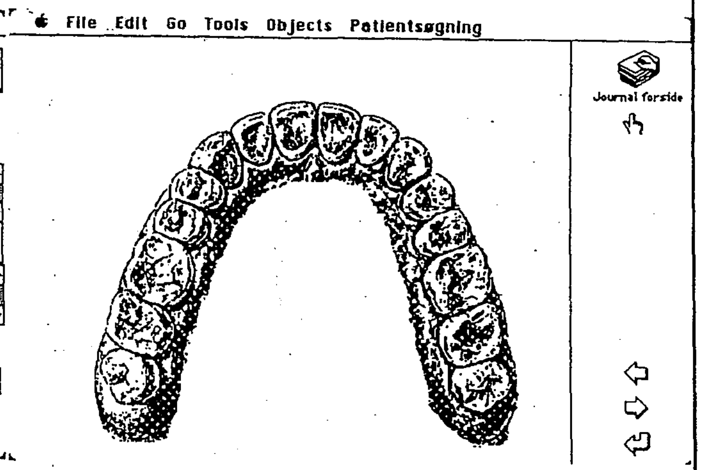


Fig. 3: Main card of Stack(3)

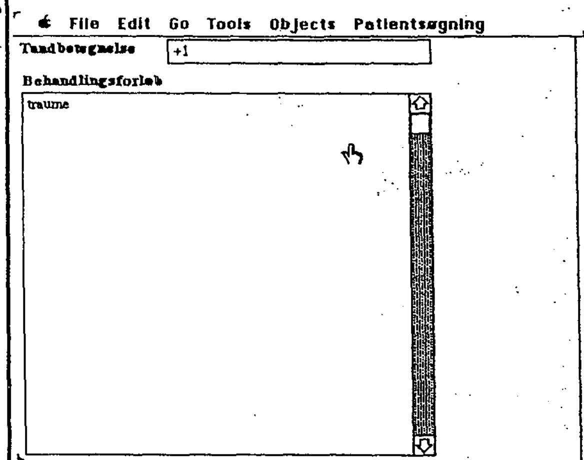


Fig. 4: Tooth card of Stack(3)and (4)

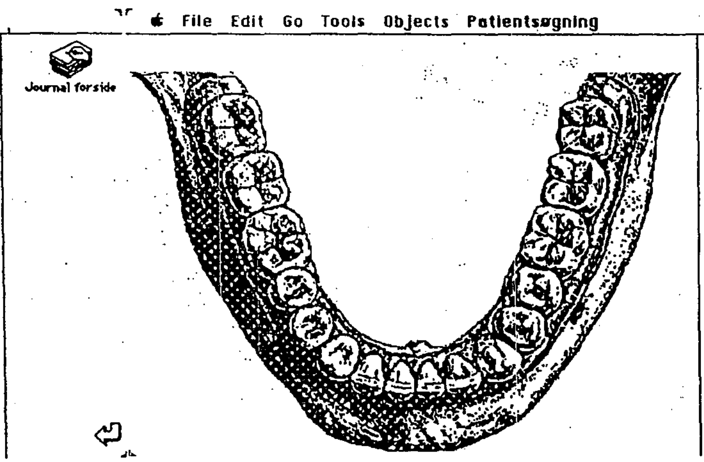


Fig. 5: Main card of Stack(4)

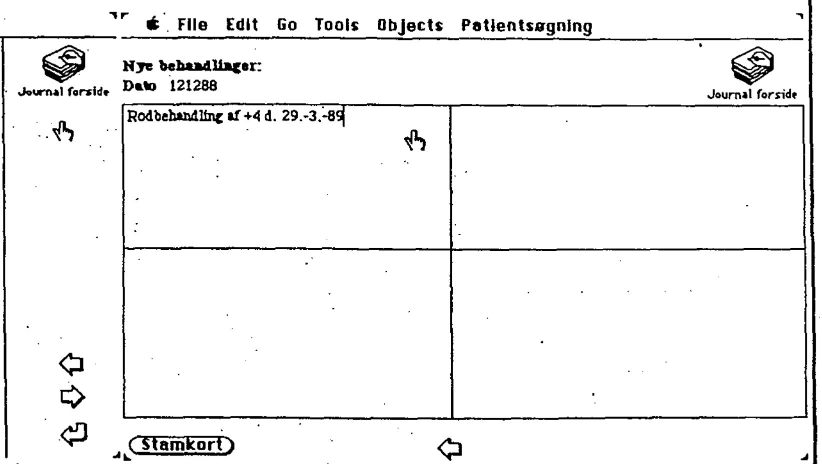
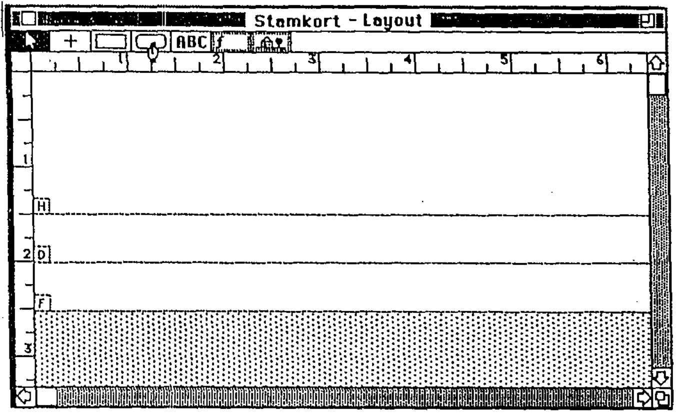
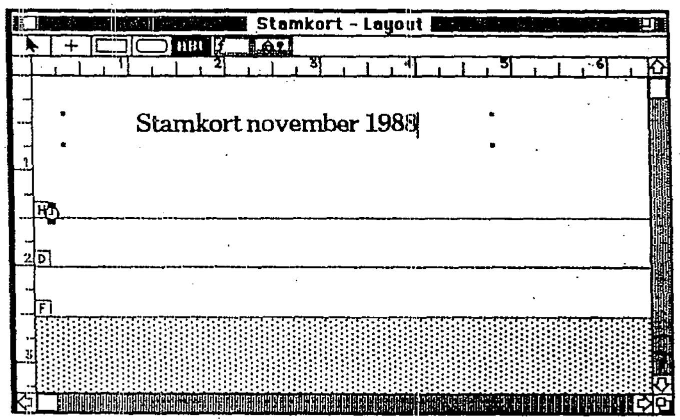


Fig. 6: Card of Stack(5)

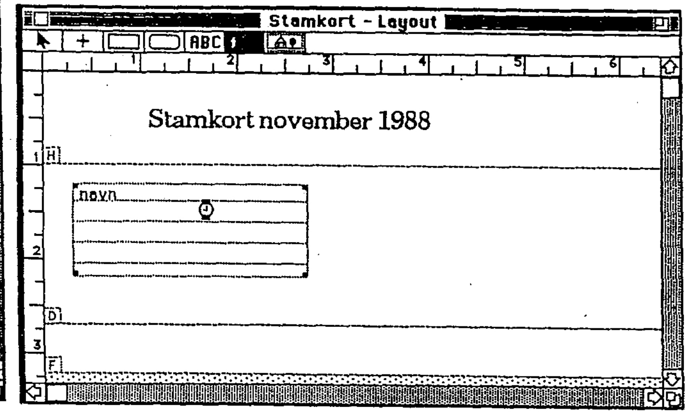
Reports™ is used in the following way: The screen image consists of a menu, some rulers and three sections, which corresponds to the header, the details or listings, and the footer of the report



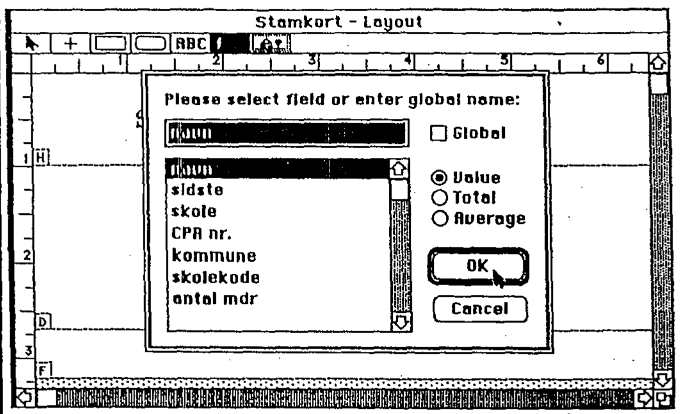
For the header section, a field is laid out into which e.g. a text can be entered.



In the details section fields can be laid out similarly, for instance a field for name:



A corresponding HyperCard field is selected from the menu:



Generating the report creates the following result shown in a previewer on the screen