Graz University of Technology
Erzherzog - Johann University

University of Music and
Dramatic Arts Graz

Institute of Electronic Music

Diploma Thesis

Spatial Auditory User Interfaces

Veronika Putz

Institute of Electronic Music
Head: Mag. DI Dr. o.Univ.-Prof. Robert Höldrich
University of Music and Dramatic Arts Graz

Graz, Oktober 2004

ZUSAMMENFASSUNG

In der vorliegenden Diplomarbeit wird das Konzept einer Benutzeroberfläche mit Hilfe von 3D Raumklang als Interaktionsmöglichkeit für blinde und sehbehinderte Anwender untersucht. Existierende Computerschnittstellen (Hardware- und Softwarelösungen) für diese Gruppe von Benutzern haben einen sequentiellen Charakter und können die Vorteile, die eine graphische Applikation sehenden Anwendern bietet, kaum ermöglichen.

Um eine gleichwertige Realisierung einer graphischen Applikation im auditorischen Bereich zu erreichen, müssen eine geeignete Codierung des Inhaltes und geeignete Möglichkeiten zur Interaktion gefunden werden. Diese Diplomarbeit beschäftigt sich zunächst mit der Frage, wie existierende Anwendungen strukturiert und semantisch sinnvoll für die akustische Präsentation vorbereitet werden können: Eine semantische Beschreibung des Inhalts von Applikationen frei von graphischen Informationen mit Hilfe sogenannter „Interaction Patterns“ wird vorgeschlagen. Diese Patterns sind Lösungsmuster für mögliche Probleme, die bei Interaktion mit einer Anwendung auftreten können, beschrieben aus der Sichtweise des Anwenders. Mit dieser allgemeinen Beschreibung kann die Anwendung in den verschiedenen Bereichen (graphisch, auditorisch) dargestellt werden. Die Transformation in den jeweiligen Bereich erfolgt mittels eines Sets aus Transformationsregeln. In weiterer Folge werden mögliche Transformationsregeln im Hinblick auf die Verwendung von Metaphoren, die Interaktion mit dem Interface und die Darstellung von Information mit auditorischen Mitteln aufgelistet und evaluiert. Das beinhaltet auch die Beschreibung von zwei Hörversuchen, die mit blinden und sehenden Anwendern durchgeführt wurden und die vielversprechende Ergebnisse lieferten. Die Auswertung der Hörversuche und die daraus gezogenen Schlüsse bilden den Abschluss dieser Diplomarbeit.

ABSTRACT

In this thesis, the concept of an user interface with the aid of 3D spatial sound as interaction facility for blind and visually impaired users is investigated. Existing computer interfaces (hard- and software solutions) for this group of users have a sequential character – they can hardly provide the benefits a graphical application offers for sighted users.

To have an auditory realisation of equal usability compared to a graphical application, a suitable encoding of the content and appropriate facilities for interaction must be found. This thesis at first deals with the question, how existing applications can be prepared for the acoustic representation in a structured and semantically meaningful way. A semantic description of the content of an application free of graphical information with the help of so-called “interaction patterns” is proposed. These patterns are possible solutions for problems and user intentions which appear when interacting with an application. With this general description, the application can be represented in different modes (graphic, auditory…). The Transformation into these modes is performed with a set of transformation rules. These transformation rules are also investigated and evaluated within the thesis with regard to metaphors, the interaction with the interface and the presentation of information by auditory means. This also includes the description of two listening tests which were performed by blind and normal sighted test users and which produced encouraging results. The evaluation of the listening tests and the drawn conclusions complete the thesis.

ACKNOWLEDGEMENTS

At first, I would like to thank my family and friends for their support throughout my studies and for their help with regard to the required sound recordings – this thesis is dedicated to them. I would like to thank Prof. Robert Höldrich and DI Christopher Frauenberger for their supervision and their advise in the past few months.

Furthermore, I would like to thank all test persons who participated in the two listening tests, especially the blind or visually impaired test users who provided me with many good ideas for the thesis and for future applications. Many thanks also go to the ISIS group (Information, service, integration and seminars for blind and visually-impaired people, www.gribus.at) which enabled the contact to the test participants.

CONTENT

LIST OF FIGURES

LIST OF TABLES

1. INTRODUCTION

This diploma thesis aims to investigate, how applications which already exist within the graphical domain or which are newly implemented can be presented within the acoustical domain by a spatial auditory user interface (AUI). The interest in an acoustical representation of applications arises for several reasons:

Spatial auditory user interfaces have the potential to ease computing access for blind users: The relevance of computer skills with regard to job qualification is known, and the impact of computers on leisure time and social life is constantly increasing. For visually impaired people, access to computers is more difficult because of a lack of efficient non-visual solutions, and it is also tied up with increased costs, compared to sighted users. Blind users have to rely on assistive software like screenreaders or on hardware solutions like braille-lines which translate the graphical information into synthesised speech or tactile information. Both technologies have a highly sequential character and, as modern user-interfaces are designed for a parallel conveyance of information, their usage is bound by slower performance and less comfort.

Many graphical user interfaces (GUIs) are overloaded with information. Supplementary to the graphical representation, auditory information could be added to reduce on-screen-complexity. Although common user interfaces include the linking of certain actions with sound events, this linking is unstructured, customisable and redundant and does not relieve the visual load – it is a multimedia gimmick rather than a source of information. Especially with respect to mobile computing and small portable devices (PDAs, cellular phones, mp3-players…), the amount of information to convey often exceeds the available screen space by

far, and users have to step through confusing context menus. The transfer of parts of the visual information to the auditory domain could reduce the visual load for both desktop and mobile applications.

Vision is often restricted by external factors, like the attention of the user being required for several concurrent tasks. As an example, the usage of commercial portable devices in public traffic can be named as well as all kind of monitoring tasks (e.g. air traffic control, monitoring of patients…). Auditory User Interfaces can be used in situations, where visual attention should not or must not be disturbed.

From these fields of applications, this thesis will concentrate on assistive technology for blind and visually impaired users, although the discussed methods and strategies are applicable to the other domains as well. The goal is to give this group of users access to the benefits a graphical user interface offers for sighted users, which have been inaccessible for them up to now. Compared to the sequential presentation which is provided by screenreaders and braille-lines, the major benefits beside others can be named as follows (after [30]):

- The graphical presentation of GUIs enables the usage of symbols and icons which makes them easy to learn and increases their usability.
- GUIs enable direct manipulation of these objects and therefore are very intuitive and transparent to novice users.
- Related information is grouped in windows, and users can organise their workspace to see and work on different pieces of information at the same time.
- GUIs enable a quick overview and quick access to the content of the interface.

A spatial AUI for blind users should be able provide these benefits within the auditory domain too.

1.1 Development and the state of the art

Up to now, among the solutions for blind users no realisation of an AUI exists which provides blind users with all of the above mentioned benefits. On the sector of private and commercial PC usage for sighted users, conveying information with auditory means is reduced to the already mentioned unstructured set of *beeps* and *plings* available in modern GUIs. Therefore, examples for existing AUI realisations are taken from the area of scientific research. The following section describes several research projects concerning AUIs with different approaches and different fields of usage.

1.1.1 Adding auditory information to graphical applications

An early example for adding *structured* auditory information to existing graphical applications is the SonicFinder by Gaver [20]. The SonicFinder, supplementary to the ordinary graphical Finder which runs on the Apple MacIntosh user interface, provides acoustical information about certain objects and actions within the interface. The system is designed as additional feedback for sighted users. What distinguishes the SonicFinder from the mentioned sounding facilities of commercial GUIs is an intuitive and structured sound mapping.

A more recent experiment for supporting concurrent graphical tasks with auditory information was published by the Naval Research Laboratory in 2004 [7]. It can be taken as example how a bad design of an AUI can break plausibility of the entire interface. In the experiment it was supposed, that for a dual task consisting of a continuous tracking activity and a second interrupting decisional task presented on two separate monitors, additional auditory information could improve the performance of test users. For the tracking task, an auditory alert was given, which varied in pitch, depending on the tracking error of the operator. Also to the decisional task, auditory information was added: The presence and the identity of a decisional event was signalled, and the location of the event was represented by spatial sound. The spatial sound location was switched between a screen-centric and an ego-centric representation for different trials. The findings of this experiment were, that the tracking error from the first task was not reduced but even slightly worsened, when the auditory cue for tracking errors was added, and that the average number of head movements was increased by it. The presence of the tracking error signal also negatively influenced the average reaction

time for the decisional task. The worst results occurred when the virtual listening spaces of the two screens overlapped.

1.1.2 Auditory user interfaces and mobile applications

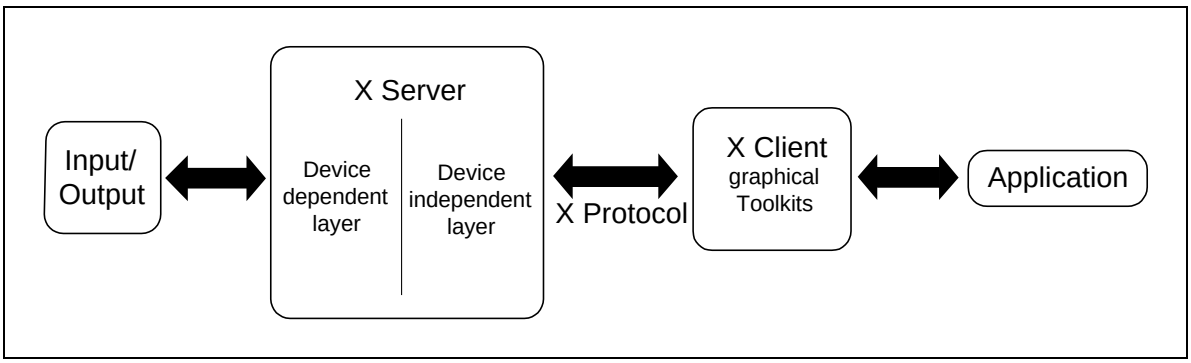
An example for the successful usage of a spatial AUI for mobile applications is given by Brewster et al in an experiment called *A diary in the sky* [43]: An auditory version of the PDA time scheduling program *DataBook* was simulated with the aid of synthesised speech and spatial sound reproduction. The application worked within the auditory domain without any visual cue: The scheduled events were presented by synthesised speech and spatialised within the horizontal plane according to their scheduling time (e.g. 12:00 is directly in front of the user, 6:00 directly behind). An experiment was conducted, where test users had to answer questions concerning a fictitious time schedule.

The test was executed with both the auditory application and a desktop simulation of the existing PDA application. The results were compared, and with regard to the subjective workload and the number of correct answers, the auditory version surpassed the visual one.

1.1.3 The “Mercator” project

One of the first approaches towards an auditory user interface framework was the *Mercator* – environment, developed at the Georgia Institute of Technology, which started in the early years of the past decade [16]. In this approach, the content of a graphical user interface is directly translated into a hierarchically structured AUI. Each GUI interface object is represented by an own sound pattern. Manipulation and interaction within the hierarchical tree is performed with the arrow keys of the keyboard. The required information for the auditory representation is extracted from the input for the graphical representation as follows: In the X Windows System, information concerning the representation of the interface content is processed from the application via X Clients to the X Server, as illustrated in [Fig 1].

The Mercator approach intercepts the exchange of information between the X Server and the X Clients to collect the information of the X Protocol which is responsible for the screen representation - a rather low-level graphical information without semantic cues. This graphical character makes the correct interpretation of the screen content difficult and provides information which is strictly graphical and partly unnecessary for the auditory interface.



[Fig 1] Client and Server architecture in the X Windows System

1.2 Approach

The examples from the previous section show, that with regard to the design of auditory applications, it is difficult to predict the usability of a layout, and that the individual representation strategy determines whether the application works or not. A spatial auditory user interface as assistive technology for visually impaired users must enable the consistent representation of arbitrary applications, therefore a general purpose representation strategy usable for any application must be found.

Additionally, an auditory user interface should be able to represent applications which already exist within the graphical domain without the necessity of rewriting them especially for the auditory representation. A strategy to automatically extract information from existing applications, similar to the Mercator – project, must be found. For this purpose, a semantic approach could be taken into account, as proposed in [17]: The information extracted from the X Window system which was used in the Mercator – project proved to be too complicated and too "graphical" for a meaningful mapping. Modern operating systems like KDE or Windows include so-called *accessibility interfaces*, which provide information for existing assistive technologies (e.g. screenreaders, magnifiers) at a more abstract and less graphical stage. The authors suggest to use these accessibility interfaces for future applications. Furthermore, a user-centred semantically based description of the extracted user interface content as input for the transformation into the auditory domain is proposed.

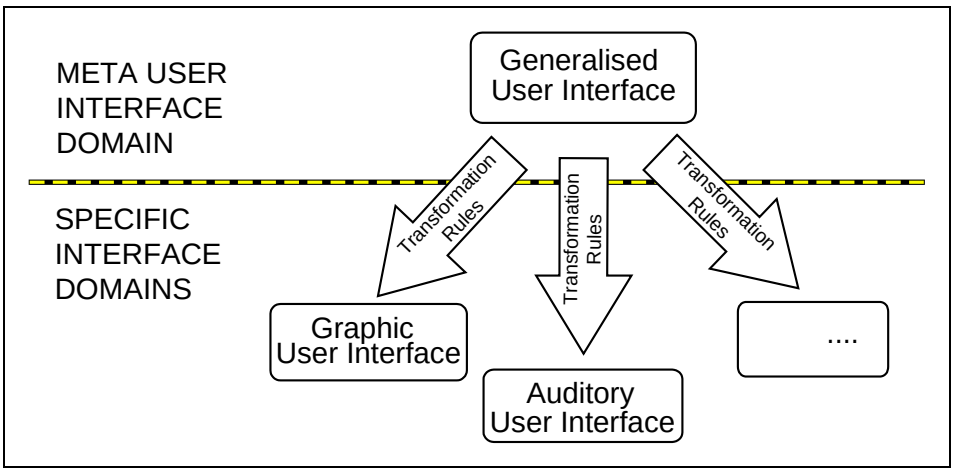
Throughout this thesis, the concept of **stepping back to a more abstract description before transferring it into the auditory domain** is kept, but the starting point is different: Firstly, a framework for a clearly defined user centred description of the content of an interface is introduced, which then can be used as input to manually design different representational

modes (visual, auditory) of new applications. This description guarantees the consistent representation of the whole application within the different modes and is the type of information which can be extracted from existing applications in the *ideal* case.

The framework can be found in chapter 2 (General User Interface Description) of the thesis. In chapter 3, existing strategies to map information into the auditory domain are investigated and summarised. The findings of chapter 2 and 3 have been tested in two different listening tests, were a rather simple and a more complex application were realised within the auditory domain. The results of the two tests are shown in chapter 4 of the thesis.

2. GENERAL USER INTERFACE DESCRIPTION

In the previous chapter, a semantic approach to prepare applications for the mapping into different types of user interfaces was proposed. For this approach, a *meta domain* which contains a general description of the application is introduced. The description within this domain is strictly semantic, without any information concerning the realisation of the application within one of the specific modes of representation (visual, auditory, haptic…). These specific descriptions are generated from the meta domain by applying a set of transformation rules to the generalised description. Each specific user interface requires an own set of transformation rules. The relationship between the meta user interface domain and the specific user interface domains is illustrated in [Fig 2].



[Fig 2] Relationship between the meta domain and several specific domains

2.1 Interaction patterns

To describe the content of an application within the meta user interface domain, it is not sufficient to simply translate all graphical attributes and relations into general ones – this would lead to a patchwork of individual elements and would contain no semantic information at all. A semantic description must include the evaluation of the purpose of all parts of the application – from the perspective of the user. The goal is to describe the application according to the arising tasks or problems of the user while interacting with it. Such a description also avoids the mapping of unnecessary information which is only required within one specific domain.

One way to do this is to define a number of patterns which describe the different interaction problems that can be solved by an appropriate representation of the application – the so-called *interaction patterns.*

The concept of interaction patterns is already known in user interface design, but there is no commonly accepted set of patterns, and generally the existing patterns are created from the perspective of the designer. A first proposal for a user-centred pattern design is published by van Welie and Traetteberg in [42]: A set of 20 interaction patterns is proposed, which is designed to solve common user problems within the graphical domain. These patterns are task-related and address different types of user problems (e.g. *the user needs to know where to find the available commands and how to activate them*), based upon several of the interaction principles suggested by Norman in [15]:

- *Visibility*: The required parts of the application should be visible at the right time and should imply correct usage. Visibility concerns the mapping between intended actions of the user and actually required operations. A generalised expression for visibility is *availability*, corresponding to *audibility* in the auditory domain.

- *Affordance*s provide strong clues to the operation of things (e.g. knobs are for turning, buttons for pushing…). When affordances are effectively used within an interface, the user knows what to do, and no further instruction is needed.

- *Constraints* minimise the number of possible actions and give additional information about the correct usage of the application.

- *Natural mapping*: If the relationship between the controlling elements of an application and their results are natural for the user, it is easily learned and always remembered. Natural mapping depends on physical analogies and cultural standards and is therefore subjective to different users.

- *Conceptual models*: By interacting with an application, the user builds a conceptual model of it. If this model is equivalent to the task model of the application, it allows the user to predict the effects of his actions. If the conceptual model is incorrect, the user acts blindly and has problems with novel situations.

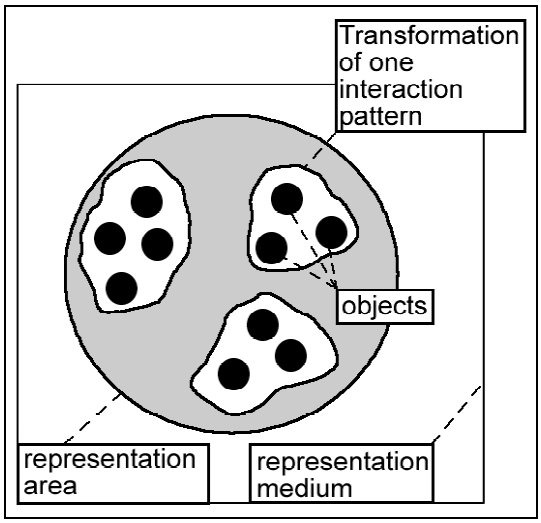
- *Feedback*: Information about the result of his actions are sent back to the user and enables immediate control of the input.

The interaction patterns from [42] build a good basis for a semantic description of the user interface content: To adapt them for a generic usage, they were uncoupled from their graphical dependencies. Thus, a set of 20 generalised user interaction patterns which enables the description of common applications within the meta domain is defined – it can be found in Appendix I of this thesis. Note that this set is not complete: For applications which cause other interaction problems, additional patterns are required. Nevertheless, the proposed patterns enable the semantic analysis of applications and their preparation for the encoding into another representational domain.

To clarify the meaning of the terms used to describe the relations between the semantic description of the interface content and the realisations within the specific domains, a terminology is given below. [Fig 3] shows how the representation area is shared between different objects within a specialised user interface domain. The objects are grouped in subsets which derive from the transformation of one interaction pattern.

- *Interaction pattern:* An interaction pattern enables the solution of one problem from the perspective of the user – it is the basic unit of the user interface description within the meta domain. Interaction patterns exist only within the meta domain and are input for a set transformation rules.
- *Transformation rules:* The generation of each specific mode of representation of the user interface content requires a fitting set of transformation rules.

- *Object*: An object is an element of the user interface which represents a certain part or unit of the user interface content. Objects exist only within one of the specific user interface domains and are created when transformation rules are applied to the generalised semantic interface description. To represent one interaction pattern, several objects may be required.
- *Representation medium*: The representation of the user interface content is bound to one or more representation media (e.g. 2D screen for graphical presentation, sound for auditory presentation, a combination of haptic feedback with sound…).
- *Representation area*: Usually, more than just one object has to be presented at the same time, so the available representation medium has to be shared between several objects. The amount of the representation medium that can be used to present a certain content is called representation area.



[Fig 3] Relationship between representation area, representation medium and objects

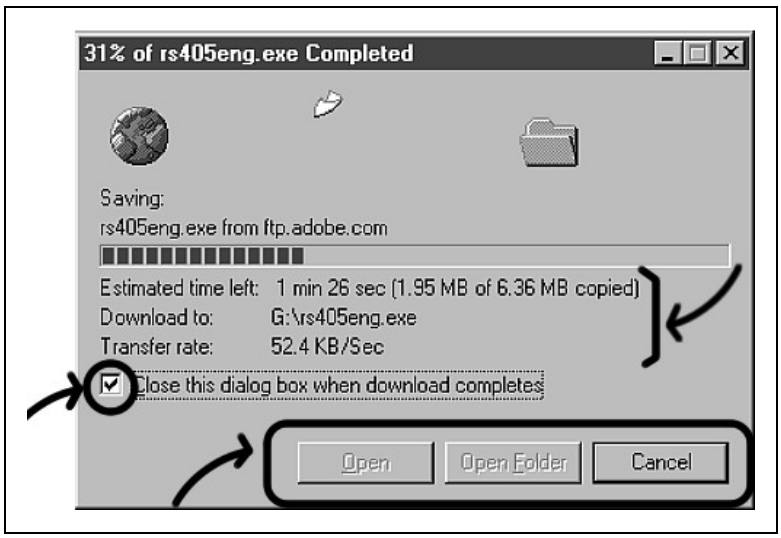
2.2 Atoms and contextual attributes

Although interaction patterns were named as the basic unit within the meta domain in the previous chapter, some of the interaction patterns are very complex structures. They consist of many parts that could be regarded as independent patterns of their own, incidentally being one module of another pattern. For an interface designer it would be very tempting to divide interaction patterns into smaller modules and to treat each of these modules as a separate pattern again – trying to find a representation for smaller units within one of the specific domains and generating the interface out of these particles. In the end, the result would be a patchwork of independent elements, reduced to their particular function with no information about the overall functionality the user is interested in – such a description would be similar to common GUI construction languages, and the user-centred philosophy would be lost.

Still, some interaction patterns consist of many smaller units which together build the pattern, and some of these units appear frequently and in different types of patterns. As an abbreviation within the meta domain, it would be practical to define a set of *atoms* which can be part of any parent interaction pattern, and it would also imply consistent representation of similar elementary units throughout the whole interface – although these units are not sufficient to solve any interaction problem. To avoid drifting towards the patchwork approach, the atoms must be connected to their parent pattern with *contextual attributes*.

To illustrate the effect of contextual attributes, a simple progress-window ([Fig 4]) is used. (The image of the progress-window is taken from [42]). Such a window, addressing the need to know whether an operation is still being performed or not and how long the execution will approximately last, is covered by the interaction pattern *progress*. In the graphical representation, the progress window includes three parts which can be defined as atoms and which are marked with arrows in the following graph: The *informational* part (the user gets information about time estimation, download destination and transfer rate), the *selective* part (a checkbox enables the user to select or deselect one option) and the *triggering* part (three buttons trigger the execution of related functions).

Furthermore, this window also conveys contextual information: The three parts mentioned above are connected to their parent pattern (the progress window) with the graphical contextual attribute of *being in the same window*, expressed by a common background colour and a surrounding frame.



[Fig 4] Progress window with atoms and contextual attributes, taken from [42]

In [Fig 4] it is obvious for the user, that all parts or atoms refer to the same function - because of the available contextual information. This additional information has to be included in the generalised description of the user interface and has to be transformed appropriately into the specific user interface domains. Without the contextual information, the connection between the atoms is lost, and the result would be useless.

The following conditions are proposed as contextual attributes:

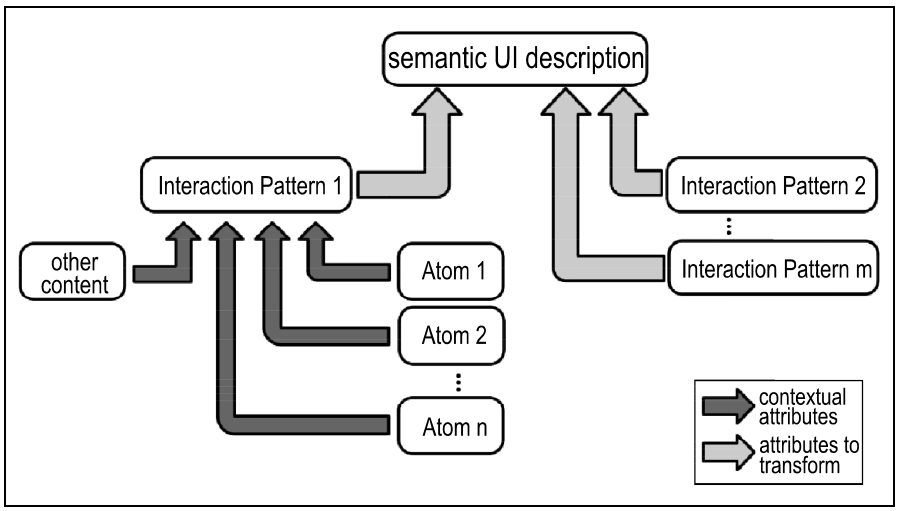
- *Similarity*: Atoms which are part of the same interaction pattern are similar to each other with regard to certain attributes, signalling equal level of importance – e.g. within textual representation, the same fontsize is used, or, taken from the progress bar example, atoms of the same interaction pattern have the same background colour. In audition, similarity can for instance be expressed with the same timbre or rhythm.

- *Proximity*: Atoms which belong to the same interaction pattern are arranged close to each other with regard to one ore more dimensions of the available representation area. For a graphical representation, the used dimension would be space. For the auditory domain, proximity could be expressed with the pitch of an auditory cue.

- *Homogeneity*: Objects which derive from the same interaction pattern are placed adjacently with regard to one ore more dimensions of the representation area. No other objects are placed between them. In audition, homogeneity could be expressed by spatial position and pitch.

[Fig 5] shows the semantic description of the content of a user interface, consisting of m interaction patterns. For interaction pattern 1, a detailed structural description is provided. Interaction pattern 1 consists of n atoms and other content which is specific to this interaction pattern only and has not been abbreviated to an atom. Each interaction pattern adds a set of attributes to transform to the semantic description. All attributes have to be transformed from the meta user interface domain to the specific user interface domains.

A collection of frequently appearing units which are suitable for building atoms with a short description of the semantic functionality of each atom can be found in Appendix II of this thesis.



[Fig 5] Illustration of the user interface description within the meta domain

2.3 Example for an interaction pattern

The structure of the generalised interaction patterns in Appendix I consists of four parts: A description of the related *user-problem*, a listing of further *conditions* which must be fulfilled to solve the problem, one possible *solution* for the problem and a listing of the *attributes* which must be mapped.

To illustrate the functionality of the interaction patterns, the *Command Area* pattern is used. It is addressed to solve an interaction problem which has already been mentioned previously:

The user needs to know where to find the possible commands and how to activate them.

When solving this problem, several further conditions have to be taken into account:

- Immediate access to all available functions of the application increases interaction speed but consumes a large amount of the available representation area.
- The amount of the remaining representational medium for the main working area should be kept as large as possible.
- Concurrent representation of many objects increases the cognitive load.
- Some functions are used more often then other functions.
- Some functions need additional parameters to be set by the user before they can be executed.

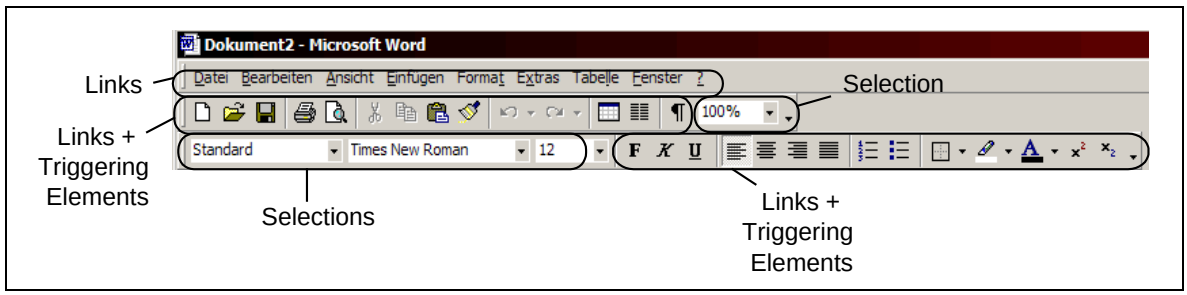
The interaction pattern proposes a *solution*:

Put the shortcuts to the possible commands in a specific recognisable area.

Then the solution is explained in detail: A part of the available representation area is reserved for shortcuts to the functions of the application. This area should be distinguishable from other working areas because of contextual attributes. If the number of shortcuts is large, they should be conceptually grouped. The command area and the included shortcuts should be accessible as direct as possible, especially frequently used shortcuts – they should be selectable by one single interaction. Feedback information about the availability of the shortcuts can be given.

The number of represented functions should not be too large to limit the required representational area and the cognitive load. Note that the description of user-problem, conditions and solutions is taken from [42].

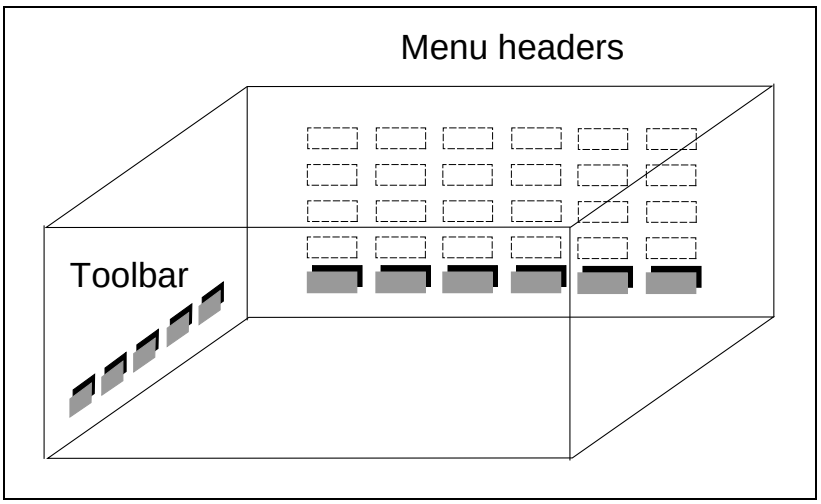
Apart from these guidelines, the general interaction patterns include a listing of the *attributes* which must be mapped into the specific domains. In our example, the realisation of the above given solution within the graphic domain will lead to a command area with menu structure and a toolbar which enables fast access to frequently used functions ([Fig 6]). Both parts include links to more detailed dialogues, parts that enable the selection of options from a limited set of elements and active areas which can trigger the execution of a desired function. These elements appear frequently and therefore they are expressed in the "link" atom, the "selection" atom and the "triggering element" atom. In the *Attributes to map* - section of the interaction patterns, the required atoms are listed. The detailed mapping of these atoms is treated in their own description.



[Fig 6] Atoms of the Command Area Interaction Pattern, graphical representation

For the second listening test, a concept to represent the *Command Area* pattern in the auditory domain within a 3D virtual room was developed. The two main parts - the menu headers and the toolbar - are situated alongside two different walls of the virtual room [Fig 7]. The buttons of the toolbar are presented with short rhythmic patterns (Earcons, see chapter 3.3.1.1 for details). The toolbar elements are connected by proximity and homogeneity with regard to space and similarity with regard to the sound design. The menu headers are presented as combination of speech and instrumental sound. The name of each menu item is spoken accompanied by an unobtrusive sound with rising pitch, starting with lowest pitch at the first header item and increasing in pitch while moving towards the last one. The menu headers are connected by proximity and homogeneity with regard to space and pitch and by similarity with regard to the compound sound design (speech + instrumental tone).

The sub menu items belonging to the menu headers are placed above the headers within the virtual room, drawn in dotted lines in [Fig 7] – for their detailed description, the *Contextual Menu* interaction pattern is required. Note, that for the second test application all of the four surrounding walls hosted parts of the interface, but only two of them were required for the *Command Area* pattern (see 4.3.6 for details).



[Fig 7] Spatial auditory representation of the command area pattern

3. MAPPING STRATEGIES FOR THE AUDITORY DOMAIN

In the previous chapter, the transformation of an application into the meta user interface domain was discussed. To create the interface representation for a specific representational mode, a set of transformation rules is required. The following chapter gives an overview of possible transformation schemes for an auditory representation.

The definition of transformation rules for the auditory domain consist of three major questions, which the following chapters try to answer:

- *Which metaphor shall be used?* The purpose of a user interface metaphor is to give the user instantaneous knowledge about how to interact with an interface - therefore the selection of a metaphor is a critical decision when designing an Auditory Interface.

- *How to interact with the application*? This part addresses the decision which input and output devices are used for the interaction. Several interaction devices are examined with regard to their usability for blind users.

- *Which design is the most effective*? The term "design" includes the strategy to convey information by auditory means and considerations regarding the spatial representation of the auditory interface.

3.1 Which metaphor to use?

According to Carroll, Mack, and Kellogg, an interface metaphor is "*designed interface actions, procedures, and concepts that exploit specific knowledge that users have of other domains*." [10]. This definition already includes the principle of natural mapping by Norman [15]: If a metaphor is chosen properly, the objects within the interface imply natural interaction. Even a novice user knows about the functionality of the objects and how to interact with them, because it is *natural* to him, and he is able to predict the effects of his actions.

User interfaces include different types of metaphorical analogies, and some of them are also used outside the computer world. A classification of metaphors due to the type of analogy is given by Barr, Biddle and Noble [3], based upon Lakof and Johnson [25]:

Orientational metaphors explain a concept in terms of space. They are not only used for navigation (*forward is to the right*), but also for quantification, which can frequently be seen in horizontal (*progress is to the right*) and vertical sliders (*up is more*) – two examples which are also valid for mechanical knobs and faders of electronic devices. In the auditory domain, pitch can be an orientational metaphor (*higher pitch is more*).

Ontological metaphors use experience of physical objects and substances to provide a better understanding: Treating time like an object is required for phrases like *I don´t have enough time*. Similar to this, treating files as objects enables us to speak of their size and their location. The ontological metaphor of considering parts of the system as objects is required for our basic understanding of computing. Thus, the same ontological metaphors can serve the visual and the auditory domain.

Structural metaphors characterise the structure of one concept by comparing it to the structure of another one. These concepts can be abstract, actual objects or events. Structural metaphors are obtained when ontological metaphors are made more specific: Instead of *X is an object*, the object is clearly specified. The desktop metaphor with the analogy between files and folders and their real world equivalents belongs to this class of metaphors. From his real-life experience, the user knows, that files can be collected in folders, folders can be labelled to define their content, and files and folders which are not required any longer can be thrown into the trash can.

Where a metaphor explains one concept in terms of another, *metonymy* is the use of "*one entity to refer to another that is related to it*" [25]. As an example, *the crown* is often used when spoken about the queen. Metonymy is frequently used in graphical interfaces when icons are used to represent applications. Also within the auditory domain, connections between auditory cues and applications or functions can be established: A simple but catchy melody or rhythm can be used to symbolise an application.

When creating new metaphors, it must be ensured, that all *metaphoric entailments* (information the metaphor implies about the application, [3]) are clearly indicated and that, whenever possible, metaphoric entailments are used to gain the advantages of natural mapping. Natural mapping and with it the quality of the chosen metaphor depends on *physical analogies* and *cultural standards* and therefore is subjective to users or groups of users. The selection of an appropriate metaphor is a very critical stage in the design of a user interface. If the number of *metaphorical mis-matches* (differences between the metaphoric entailments and the functionality of the application) shall be low the possible actions and interactions of the user interface are limited by the metaphor. Edwards and Mynatt [31] after Carroll, Mack, and Kellogg [10] defined three types of metaphoric mis-matches:

- The metaphor may contain concepts and actions that are not supported by the computer application. The user will try to perform actions that lead to failure.
- The computer application may support actions that are not present in the metaphor. Actions that are not explained by the metaphor will be more difficult for the user to learn and remember.
- The computer application may behave in ways which directly conflict with what the metaphor predicts. This type of mis-match is the worst since it will cause the user to mistrust the metaphor as a whole.

The classification of metaphors given above includes many metaphors which have a rather graphical approach, and especially many orientational metaphors like *progress is to the right* and *up is more* are commonly established in graphical computing. For the auditory domain, comparable metaphors have to be found. Up to now, the majority of realisations of auditory interfaces exploit mainly one structural overall concept comparable to the desktop metaphor. The following section is a short summary of metaphors used for implementations of AUIs:

The *Audio Rooms* metaphor used in the Mercator – environment, described in [31], uses a three-dimensional layout of rooms as container for the grouping of applications and data. Each room can contain files or applications, and a doorway leads to other rooms. Within a room, the objects are placed two-dimensionally. They are manipulated by first selecting and then applying an operation onto them. Different room characteristics like floor covering, windows and wall-papers are not implemented. The navigation within 3D rooms is of daily experience for blind and normal sighted users as well and therefore can be regarded as common knowledge, although congenitally blinds have minor impression of spatiality. The Audio Rooms metaphor includes some mis-matches, like the dynamically changing size of rooms or the ability of the user to teleport from one room to another without crossing all rooms that lie in between. The metaphor also has the disadvantage, that no room can be created within another room, so the possibilities for a hierarchical structure are limited.

A more abstract metaphor called *Acoustic Bubbles* was developed by Sawhney and Murphy [37]: Ambient spaces or bubbles are immersed in a fluid acoustic medium which provides a navigable environment. Each bubble has a unique audio texture and can encapsulate other bubbles, creating a hierarchical relationship. A user can enter a bubble without doorways through translucent walls and is immediately positioned at the centre of the bubble. Within the bubble, the user has six degrees of freedom to navigate. The bubbles can be collapsed, blown-up or merged based on the content. For this metaphor, no mis-match can be found on the first sight, but the concept and the possible interactions within the bubbles surely require more explanation because the metaphor is rather abstract and exploits less common knowledge.

In an application called *Panphonem*, described in [34], a well is used as room model for a 3D audio interface which means, the virtual room has a cylindrical shape. The well hosts several social tasks (email, phone calls…). In each level of the well, several different tasks are placed, according to their different level of privacy. Although in *Panphonem* a real object, the well, is used as model for the structural layout of the virtual room, it can hardly be counted as metaphor, because the type of the room layout has no implication on the possible interactions and the functionality of the included objects.

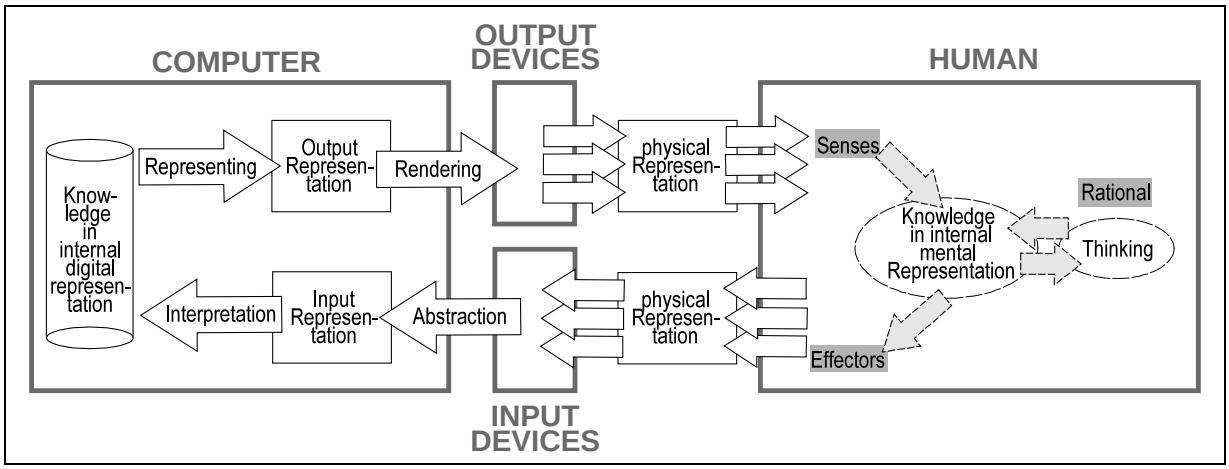
A *Clockface metaphor* is used in an auditory version of the PDA-program *DataBook* by Walker, Brewster and McGookin [43] which is already mentioned in the introduction of the thesis. The layout is horizontal and circular, a slice around the head of the user. This slice corresponds to a clock-face. The sound information is spatialised and presented via head-phones. Thus, temporal information can be derived from the position of the sound (e.g. 12:00 is expressed by a sound right in front of the user). The subjective ratings of the test users were very well, and many participants stated that the audio condition required less interpretation than the visual task, because the temporal information came "for free" from the spatial sound position. The Clockface metaphor is an excellent example for the benefits the choice of a suiting metaphor can bring for certain applications.

These examples of metaphors are addressing the usage of one "*over-all*" structural metaphor and its entailments rather than trying to find auditory equivalents for the many orientational metaphors included in GUIs. One investigation in a mapping comparable to the *up is more* metaphor shall be named: An experiment about psychophysical scaling and sonification of data has been carried out by Walker and Lane in [44]: Blind and normal sighted participants made magnitude estimates of temperature, pressure, velocity, size, and an amount of money according to changes in pitch, tempo and timbre. The suggested mappings of the participants were evaluated with regard to polarity and slope. In most cases, the polarity was the same for both sighted and visually impaired participants, although there were some exceptions where the two groups suggested different polarities. For the slopes, there was general agreement between the groups, but the slope of the scaling functions depended on both the sound attribute that was varied and the type of data that the sound was supposed to represent.

This experiment shows that for auditory user interfaces, powerful orientational metaphors can be found although much further experimental work will be required, and that the design must take into account whether the user is visually impaired or not. The equivalency with regard to usability between GUIs and AUIs will strongly depend on the establishing of these metaphors within the auditory domain.

3.2 How to interact?

A human computer interaction process is defined as the exchange of information between user and computer over one ore more physical media, depending on the available in- and output facilities [12]. The computer is connected with these physical media through I/O devices which are responsible for evaluating the user input and rendering the computer output. Generally, computer input and output requires a two-step transformation: Abstraction and interpretation for the input, representation and rendering for the output. For the processing on the human side, different models exist. In [12], the following scheme is used: Information is perceived through the senses, exported through effectors and is internally processed and stored in distinct mental representations. [Fig 8] illustrates this connection:



[Fig 8] Model for human computer interaction, after [12]

For the performance of input actions by the user, again the principle of natural interaction is important: The more natural the physical interaction is, the easier it will be for the user to concentrate on the *things he wants to do*, instead of thinking about *how to perform them*. A selective operation with an input device which enables the user to perform some kind of grabbing movement (e.g. a data glove) does not need much interpretation, because it is equivalent to the real-world movement. It is natural for (normal sighted) users to point at objects which are distributed in their real world 3D environment, but to do this within the two-dimensional space of a PC screen with a mouse as intermediate device already requires some practice. Thus, natural interaction is limited by the available interaction device.

In [22], aspects of successful interaction with auditory interfaces is extrapolated from human interaction with musical instruments by Hunt and Hermann:

- Like in interaction with musical instruments it is important, that the given feedback is real-time or with low latency to enable the user to relate actions and reactions.
- Physical or tactile interaction is required. A gestural control without physical interaction is more difficult and does not occur within the real world.
- By accepting increased learning times, increased subtlety and complexity of the performance can be yielded.
- The interface must react in a well-known, natural way, e.g. when something is hit harder, it should sound louder.
- Information, if distributed over different modalities (visual, acoustic, haptic), must be conveyed coherently.

As assistive technology, interaction devices which can be utilised by blind and normal sighted users have to be found. In this thesis, the focus shall lie on interaction devices which are inexpensive, compared to existing solutions like a braille-line or a screen reader. The additional costs for equal access to computers for visually impaired users should be minimal. Furthermore, all required components should be available in a customary PC shop and also be supported by a conventional PC system. With this restriction, the range of input devices shrinks to three possibilities: keyboard, mouse and joystick.

3.2.1 Keyboard

Customary keyboards can be utilised by blind users very well because they are truly tactile devices. They have no problem to orient themselves on a keyboard and usually perform a large amount of their interaction work with keyboard shortcuts. The only required help are the small palpable dots or bars on two of the keys to ensure the correct positioning of both hands – something which is also required for efficient usage by sighted users. The limitations of keyboard shortcuts are, that not every function is provided with shortcuts, and that the shortcuts differ between applications and therefore have to be learned by the user. To completely control an application with the aid of a keyboard, the user needs a lot of in-advance-knowledge of it – then he can work independent of the graphical representation.

The benefits of user interface concepts like two-dimensional movements and natural interactions like *drag and drop* can hardly be exploited with a keyboard, because navigation with arrow keys can only be an insufficient replacement of a real two-dimensional movement.

3.2.2 Mouse

The mouse as input device can hardly be used by blind people – the movement of the device gives no feedback about the position of the mouse pointer. When it is lifted from the pad and moved without contact to the surface, the virtual mouse pointer is not affected. The blind user has no feedback, whether he has moved the pointer two centimeters or two meters. Although the on-screen mouse pointer stops when the borders of the screen space are reached, the real world device can be moved endlessly. Additionally, the velocity of the movement can be customised, which is another unknown parameter for a blind user.

An experiment concerning the usability of a mouse for auditory interfaces can be found in [33]. In this test, the target acted as sound source and the cursor as microphone. Thus, the distance between cursor and target determined the amplitude of the perceived sound. Different types of sound representation (mono or stereo), target positions (linear or two-dimensional) and distance coding (linear dependence on the distance or artificially expanded) were used. The best results were reached within the linear layout for stereo presentation with expanded dynamics. Although the achieved error rate was relatively small, the results were poor with regard to working speed, compared to the required time to perform the same test with visual feedback. The test was performed with a group of normal sighted test persons. It can be assumed, that with visual impaired users the results would have been even worse, because unlike a sighted test user, a blind user would have had no feedback of his movement within the real world, and performing a linear movement would have been more difficult for him.

3.2.3 Joystick

Like a mouse, a joystick generally allows movement on a two-dimensional plane, but the possible interaction range of a joystick is bounded mechanically to a sphere around a central position. The angular position of the handle can be used to estimate the absolute location of the virtual pointer on the screen. Furthermore, if the handle of the joystick is released, it returns to a central position, and relocation is simple – if orientation is lost, it is easy to return to a basic position, and for this returning movement feedback is given.

On a joystick several buttons can be found, which can host all functions a mouse usually enables: clicking, right-clicking, scrolling… Many joysticks also provide of a so-called "thrust-wheel" – it can be used for navigation in the third dimension. An investigation concerning the usability of a joystick for visually impaired users to point in a 3D virtual environment was carried out in the second test described in chapter 4 of this thesis, and the results are encouraging.

3.2.4 Other devices

Apart from the three options mentioned above, many other less common devices could be used to enable 3D interaction for blind users. Most of them were developed for interacting with visual virtual environments (VE). A taxonomy for these devices can be found in [18]. If VE and 3D interaction can be established for ordinary PC usage, the related interaction devices will become common PC equipment and thus less expensive. Therefore some prominent VE interaction devices are discussed in this chapter shortly:

Data Gloves can create the impression of interacting directly through the hands without an interaction device and enable natural interactions like grabbing and pointing. Furthermore, recognition of gestures and 3D position tracking is possible which enables a variety of input interactions. As drawback, the lack of physical boundaries of the interaction sphere must be named. The same goes for *3D sticks* or *wands.* They can be used as pointing devices and are in fact 3D mice which enable a movement with six degrees of freedom. Additionally they can host several buttons for selection. Devices which support interaction bounded to a certain sphere like a joystick exist also within the domain of VE interaction. The *elastic general purpose grip* or EGG is an egg shaped object which is elastically mounted on rubber ribbons in the centre of a quadric frame. Interaction takes place by physically moving the object within the frame. Although it is used for rate control rather than position control within VE applications, it could also be used for 3D spatial interaction with an AUI. Compared to a small and portable PC mouse, the EGG appears to be clumsy and impractical. A *Graphic Tablet* supports a 2D interaction on a limited rectangular area. Small versions of these tablets can be found on laptops and PDAs as replacement for the mouse. Input takes place with a special pen or simply with the fingers. For the usage as interaction device for blind or visually impaired users, a bigger tablet than the laptop version will be required to host several active areas at the same time, and a solution for the movement in the third dimension must be found.

3.3 Which design is the most effective?

This chapter includes strategies to convey information with auditory means, strategies to represent this information in a limited representational area or to present sound concurrently, and the effects of adding spatial information, distance coding and room information to the interface.

3.3.1 Strategies to convey information with auditory means

Some guidelines for the auditory representation of information were proposed in [4]:

- From auditory cues, listeners are much more capable of perceiving changes than to make absolute, quantitative judgements.
- Pitch, duration and rhythm can be discerned and can be used to carry information.
- Information coded with different levels of amplitude is difficult to catch.
- Multiple mappings (mapping of more than one sound attribute to one dimension of the data) can be useful.
- The dimensions of sound (amplitude, pitch, timbre) are not orthogonal, therefore careful design is required to avoid destructive interactions

In Literature, five different ways to represent information acoustically can be found: Earcons, Auditory Icons, Filtears, Continuous Sound and (synthesised) Speech. The following section offers a short description of these options with their advantages and limitations. The auditory representation of an application includes different types of data which have to be represented (absolute data, relative changes, background activity…) – each of the five ways mentioned above fits to a certain type of data, therefore, the best strategy to convey information within the auditory domain is to use a combination of all these strategies.

3.3.1.1 Earcons

The concept of Earcons was introduced by Blattner, Sumikawa and Greenberg in [41]. An Earcon is a short rhythmic or melodic pattern used as an abstract symbol for a piece of information without intuitive link. Like a foreign language, the meaning of the Earcons must be learned by the user.

Attributes of Earcons can be varied with regard to pitch, timbre, rhythm, dynamics or register. Longer, context dependent information can be created by serial or parallel combination of several Earcons (compound Earcons) or by using a hierarchical tree structure, organised according to the structure of the represented information, where the leaves inherit properties from their branches.

Advantages:

- Representation of hierarchical structures and parameterisation is possible.
- Once a basic vocabulary is learned, new words can be added which will be easily memorised by the users.

Limitations**:**

- No natural mapping is exploited: The meaning of each Earcon has to be learned and memorised which results in an increased mental load.

3.3.1.2 Auditory Icons

The concept of Auditory Icons was developed by Graver [19] – based upon the theory, that people do not describe sounds according to their primary attributes (pitch, duration, timbre), but try to identify the source which has caused the sound. Auditory Icons are short recorded samples of natural environmental sound and less abstract than Earcons. They are mapped to the conveyed information intuitively with a semantic relationship to the represented object (e.g. the sound of a police siren could be used as alarm signal). An example for the usage of Auditory Icons is the SonicFinder [20]. More detailed information about design aspects of Auditory Icons can be found in [30], where the design process of Auditory Icons for the Mercator- Project is described.

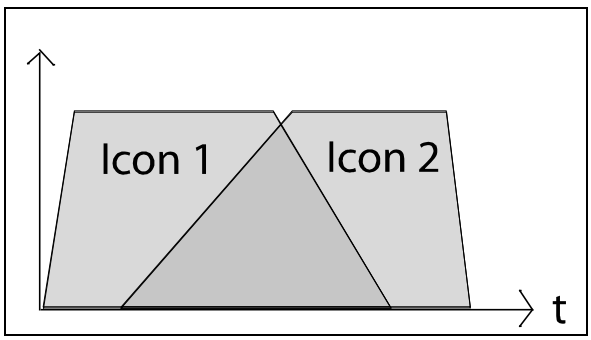
Advantage:

- Natural mapping is possible: Because of the semantic link, the user will know the meaning of the icons without detailed explanation, and the icons will be easier to memorise.

Limitations:

- The parameterisation of natural sound samples to express relative size is difficult.
- Users must be able to recognise the sound – one sound sample can cause many different interpretations by different users.
- The mapping of the sound must be intuitive – this increases the effort for coding larger amounts of information. Cultural and educational factors determine how intuitive the mapping is perceived by each user.
- It is difficult to find Auditory Icons which all fit to one metaphor, and adding new Auditory Icons to an existing set can be difficult and confusing.
- The usage of Auditory Icons can easily be annoying

For Earcons, the ability to create an abstract language to express hierarchical structures is a great benefit. An approach to enable these structures for Auditory Icons as well, a so-called *iconic language*, was used for the first listening test which is described in chapter 4 of this thesis: The idea is to pack more than just one piece of information into one icon. For instance, the icons for the confirmation of dialogues with OK or Cancel are overlapping combinations of their general sound followed by an icon which represents the dialog in which the confirmation is placed in. Thus, the confirmation icons include information about their own functionality and about the current context. The amount of temporal overlap of the two icons [Fig 9] depends on the sound structure of the used icons. For the test, the combined icons were mixed manually.



[Fig 9] Iconic language: temporal overlap of two Auditory Icons

3.3.1.3 Filtears

Filtears are auditory cues manipulated by filters and can be used to convey additional information by an auditory cue without altering its identity. The concept was developed by Ludwig, Pincever and Cohen [27]. With Filtears, different states of an auditory item can be coded. Some Filtears are suggested by Ludwig et al:

- *Self-animation*: Non-linear wave-shaping produces a more excited sound which can be used to accentuate a sound item compared to the others
- *Thickening*: A chorused doubling effect thickens the sound
- *Peaking* amplifies the signal
- *Distancing*: Reverberation and echo create distance information
- *Muffling*: linear, low-pass filtering creates impressions of confinement or distance
- *Thinning*: linear, high-pass filtering makes a sound item thinner

Filtears enable a dynamical coding of different states (e.g. “selected”, “enabled”, “disabled”) and can therefore be combined with the other encoding possibilities. One drawback of the above mentioned filters is that, by modifying the original sound, they can alter intelligibility.

3.3.1.4 Continuous Sound

For data changing over time (e.g. background activity) or to convey the presence of persistent objects, continuous audio or *sound textures* can be used. Continuous background noise can also be used to inform about the current location within an environment or as unique signature of a certain environment. For this purpose, unobtrusive music can be used [37].

3.3.1.5 Synthesised Speech

The spoken representation of information is a widely used way to provide computing access for visually impaired users. But, as stated by Brewster [6], speech has a serial nature and therefore is slow. To comprehend the message, the listener has to hear it from the beginning to the end. If the reproduction speed is increased, the recognition rate decreases, although visually impaired users reach higher recognition rates than users with normal seeing ability. Especially synthesised speech is limited to a rate far below reading speed and is a heavier load for short time memory than natural speech [40]. Therefore, if speech is used to convey information within an auditory interface, it is important, that the textual information is reduced to short and catchy keywords.

Speech cannot be used to convey monitoring information: Because of the big dynamical range, *habituation* (continuous sounds drift into the background of consciousness after a short period of time) is difficult to achieve for speech [9]. Additionally, background speech is more disruptive than non-speech sounds. This is also valid for vocal music [2].

Advantages:

- Speech is the most effective and least ambiguous way of conveying information acoustically.
- With the ability to generate speech synthetically, the coding of information via spoken language is the most flexible option with regard to a dynamically changing content of applications.
- Speech can be used to convey information without encoding – the meaning of spoken icons does not have to be learned by users.
- Presentation of absolute data is easy.

Limitations:

- Speech is very attention-grabbing, the usage must be reduced to keywords.

3.3.2 Concurrent sound reproduction

The options to convey information with sound in the previous chapter were discussed with regard to an exclusive representation only. But if a whole application has to be represented by auditory means, it might be necessary to play several sound items concurrently. In this case it is crucial, that all items remain distinguishable, otherwise the user can not extract useful information. The sensitivity of the different means to convey information with regard to concurrent sound reproduction is investigated in this chapter.

Brewster [6] states, that from the total amount of received auditory information, humans tend to sort sounds not with regard to pitch, intensity and timbre, but rather in terms of discrete sound sources or *streams*. Each stream can have its own pitches, intensities and timbres. He (after Bregman [5], Williams [46] and Deutsch [14]) also offers a collection of perceptual and physical factors by which sounds are grouped into one single source or segregated into separate different sound sources:

- *Similarity and dissimilarity:* Sounds with share the same attributes (pitch, timbre, loudness, spatial location) will be perceived as related.

- *Proximity*: Components which are close in time and frequency will be related.

- *Continuity:* Sudden changes in pitch, intensity and spatial position indicate that new sources have appeared. Different musical instruments playing in unison can be differentiated when their attacks occur with a temporal delay and if they have different temporal envelopes [35].

- *Coherence*: Components of one source vary coherently, e.g., they change their intensity and pitch together.

- *Fundamental frequency*: If two sounds with different fundamental frequencies are combined, they are perceived as two sources, and if they have the same fundamental frequency, they are heard as one source. This is in contrast to the results in [36] by Reuter: Two instrumental tones with the same fundamental frequency and with removed temporal cues (attack times, transients) can still be differentiated as long as they have different formant areas.

- *Rhythm*: Rhythmic patterns tend to be perceived as coming from the same source.
- *Scale and key structure:* People can judge the differences between two patterns of tones according to their knowledge of scale and key structures.
- *Spatial separation* of two sources increases the ability to correctly segregate them. The impacts of spatial separation are discussed in the next chapter.

With reference to these factors, the stability or sensitivity of Auditory Icons, Earcons and Speech to a concurrent representation *without* spatial information (i.e. the concurrent sources are not positioned at different locations) is investigated. The effects of spatial information on source segregation are addressed in the next chapter.

Concurrent presentation of Auditory Icons

As Auditory Icons are recordings of different natural sounds, one important prerequisite for source segregation is fulfilled per definition: Auditory Icons originally *come* from different sources. Furthermore, several Auditory Icons are not very likely to have the same fundamental frequency, the same harmonic structure and the same temporal envelope. Nevertheless, the selection of the Auditory Icons must ensure that they sound as differently to each other as possible to make them distinguishable – for sequential and for concurrent representation.

Concurrent presentation of Earcons

Earcons are abstractly composed and can therefore be designed to fit for a concurrent representation. Still, due to their synthetic character, Earcons are less resistant to errors caused by concurrent presentation than Auditory Icons. A test concerning the concurrent presentation of Earcons was carried out by McGookin and Brewster in [28]. In the test, participants had to identify Earcons within a group of three or four concurrently presented Earcons. The test showed, that the identification of multiple concurrently playing Earcons is difficult, and that not more than two Earcons could be identified correctly. Slight improvements were gained by presenting each Earcon with a unique timbre and by staggering onset times.

Concurrent presentation of speech - the cocktail party effect

The human ability to pay attention to a single talker and shadow competing talkers or background noise is known as *Cocktail Party Effect* [1]. Although it is not completely clarified which cues contribute to what extent to this effect, it is known to be a combination of what is called low-level grouping by the factors that also influence segregation of non-speech sounds and scheme-based segregation on a higher level of central processing.

With regard to scheme-based cues, experiments have shown that segregation is improved if the listener is consciously listening for something. In an experiment by Cherry [11], artificially constructed "probable" phrases were nearly impossible to segregate compared to the same experiment being carried out with meaningful sentences. Obviously, the performance of selective listening depends on information rather than on physical stimuli, or in other words, the contextual probability of a word appearing in a message influences the probability of correct source segregation.

For low-level-grouping, important factors for segregating speech sounds are spatial separation (see next chapter) and differences in the speaking voices [1]. The cues available in natural speech signals are *pitch trajectories* (the pitch of a speakers voice changes slowly and follows melodies which depend on grammar and language), *spectral continuity* (formants of successive utterances change continuously), *fundamental frequency* (if synthesised speech sounds are presented at the same fundamental frequency, segregation is more difficult) and *harmonics* (speech harmonics change parallel, when the pitch of an utterance changes).

An evidence for the influence of low-level grouping on speech can be given: An experiment considering the effectiveness of multi-talker speech displays with a special focus on military communication is described in [8]. One task of the test investigated the intelligibility rate of the spoken messages while increasing the number of concurrent speakers from 0 to 3. When the test was carried out with different talkers of the same sex, each competing talker reduced the performance by a factor of 0.6 respectively. Higher recognition rates were reached when the same test was carried out with two talkers who were different in sex. Although the experiment was performed with recordings of natural speech also for synthesised speech the suggestion can be made, that different synthetic voices with different fundamental frequencies should be used to present several items at the same time.

The same experiment also investigated the influence of adding spatial information to the test layout. In the case with one interfering talker, spatial separation increased performance by approximately 25 percentage points, in the cases with two or three interfering talkers the percentage of correct responses was nearly doubled.

3.3.3 Effects of adding spatial information and room information

Important features of AUIs definitely are spatiality and the impression of being in a real room. But to what extent can a spatial distribution of competing sources influence source segregation? Can it be used to convey information? Additionally to the spatial positioning, distance information and room information can be added to the sound sources - what impact does this information have on the interface? Before trying to answer these questions, a short overview over spatial listening and room and distance coding is given in this section.

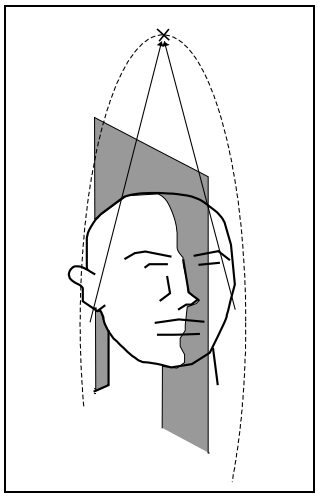
Spatial hearing

For spatial hearing, a combination of monaural ("with one ear") and binaural ("with both ears") cues is responsible. The two binaural cues or time-domain-cues are the Interaural Time Difference (ITD) and the Interaural Intensity Difference (IID):

For low frequencies, the ITD is the principal localisation cue: Sources are localised by noticeable time and phase differences of the incoming sound wave between the two ears, depending on the angular position of the sound source to the median plane. For mid and high frequencies, the wavelength of the incoming sound wave shrinks to a size comparable to the diameter of the human head (app. 17 cm), and the phase information becomes ambiguous – this happens at frequencies higher than approximately 1500 Hz.

For high frequencies, localisation is mainly performed according to the IID: The incoming sound wave is shaded by the human head, and the intensity difference between the "direct" and the "shaded" ear determines localisation. For low frequencies (below ≈ 300 Hz), this effect does not occur, because the wavelength of the incoming wave is large compared to the diameter of the human head and gets diffracted around the head - thus the perceived intensity is equal for both ears, and only temporal cues remain.

In addition to time and intensity differences, monaural cues or spectral cues exist: For an incoming sound wave, the system of outer ear, head and torso acts like a filtering process with a direction-dependent amplitude and phase behaviour. This monaural or spectral cues are required to explain the human ability to localise sound on the so-called *Cones of Confusion*: The median plane is a fictitious plane vertically drawn through the nose of the listener. As every point on a circular path on this plane has equal distance to both ears, IID and ITD can give no information for localisation [Fig 10]. Still, the perceived sound changes when the source moves on this circular path.



[Fig 10] Median plane and cone of confusion

For the filtering effect, no analytic solution can be given, but it can be simulated for a sound spatialisation via headphones by a set of impulse responses (HRTF or *Head Related Transfer Functions*), measured in a certain angular resolution on test subjects or on a dummy head. Within these measurements, binaural and spectral cues are included. As size and shape of outer ear, head and torso differ individually, the usage of non-individualised HRTFs is an approximation of the real listening experience.

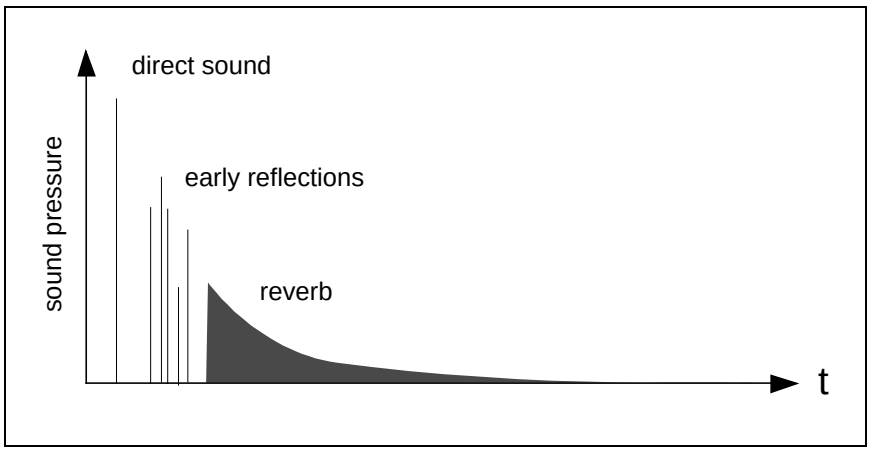
The errors caused by non-individualised HRTFs, described in [45], are *localisation errors* (or a deviation of the perceived to the encoded direction), *front-back-confusions*, a *localisation blur* (or an increased width of the perceived direction) and an *externalisation error* or an inside-the-head-localisation of the sound source. Localisation capabilities are improved by small subliminal head movements – therefore, the performance of sound spatialisation with HRTFs can be improved by using a head-tracking system.

Room and distance information

Within natural environments, a sound source is not only perceived by the sound which travels directly from the source to the listener, but also by the reflections of the sound source on surrounding surfaces [Fig 11]:

Early reflections are strong reflections of the sound source which appear shortly after the direct sound (< 50 ms) – they increase the energy of the direct sound source without altering the perceived source position and improve the localisation of sound sources. Early reflections can be calculated with the use of geometric acoustics by creating image sources. With increasing time, the density of reflections increases while their energy decreases exponentially – the so-called *late reverb* is usually evaluated in statistical manner or energy measurement and creates an impression of "being in a real room". Late reverb does not contribute to localisation and influences localisation accuracy negatively.

Direct sound, early reflections and late reverb are used to classify acoustical properties of rooms with energetic measurements such as reverberation time (T_{60}, the required time to reduce sound energy by 60 dB in [s]) and early decay time (EDT, the required time to reduce sound energy by 10 dB).



[Fig 11] Direct sound, early reflections and late reverb

For the perception of distance, monaural cues are responsible:

The sound pressure of a sound source which arrives at the location of the listener is determined by the distance r between source and listener. For anechoic spaces and spherical waves, the sound pressure amplitude decreases with 1/r. In echoic spaces, the decrease is less than 1/r. Sound pressure can be used to perceive relative changes of distance.

Depending on the distance, the timbre of a sound source is changed, because higher frequencies are shaded during the spreading of sound within air. To judge distance according to timbre, a-priori knowledge of the sound source is required.

In echoic spaces, distance information can be extracted from the relation between direct sound, early reflections and late reverb. If the direct sound is small compared to the late reverb, the impression of distance arises.

3.3.3.1 Binaural unmasking

The positive influence of available spatial information on source segregation is known as binaural or spatial unmasking and contributes to the Cocktail Party Effect. If two sound sources are played concurrently and additional spatial information is available, it is easier to pay *selective* attention to one of the sources, if their spatial positions are different. Experiments describing this effect can be found in [11]: For test listeners it was easy to pay attention to one source while rejecting the other one, if the concurrent sources were presented to different ears. The amount of improvement caused by spatial separation increases with the spatial difference.

Similar results can be found in [38] for *divided* attention, where the listener has to pay attention to two sources at the same time, although the improvement which can be gained by spatiality is bigger for selective attention than for divided attention. In the same study, experiments concerning the importance of knowledge about the source locations brought the result, that performance of the test users decreased, when source positions were changed randomly. The best performance in divided attention tasks was achieved, when one of the sources was remarkably louder than the second one.

3.3.3.2 Can spatial information be used to convey information?

The experiments concerning the Cocktail Party Effect show, that spatial information definitely *helps* to segregate different sources which are presented concurrently. But is spatial information *sufficient* for source segregation, or, in other words, can it be used to convey information itself? To answer this question, a closer look to object-building and figure-ground-segregation is taken. The prerequisites for building auditory and visual objects are explained in [23]:

Vision primary deals with surfaces that reflect light, what lies behind those surfaces usually is occluded. Vision further deals with location and colour of the surfaces. *Audition* primary deals with emitting sources, other sources are not generally occluded, although masking effects exist. Audition further deals with reflections caused by surrounding surfaces.

In both domains, objects are built by figure-ground segregation. The result of visual grouping is a *figure*, its equivalent within the auditory domain is the already mentioned concept of an *auditory stream*. The authors state, that "*any object, be it visual or auditory, must have a boundary*". To find possible edges to define these boundaries, the *theory of indispensable attributes* [24] is used: An attribute is indispensable, if and only if it is a prerequisite for perceiving more than one source.

Boundaries for object segregation can be found with regard to this attribute. According to [23], for vision, dimensions with indispensable attributes are space and time, the colour of objects is none. For audition, objects are formed with regard to frequency and time, space is not indispensable, which means, that in the absence of edges in the dimensions of frequency and time, space alone is not sufficient for source segregation, although it *gives further aid* in segregating objects. A counterexamples to this statement is given by Darwin [13]: If a sine-wave is presented to both ears with equal amplitude and a pulsing increment of the same signal is added to one ear only, two signals can be separated: A pulsating signal on one ear and a steady tone perceived in the middle of the head. In this case, signal onsets are used for source segregation.

As conclusion it can be suggested, that, although the role of spatial information with regard to auditory streaming is not completely clarified, the coding of information with spatial cues alone is not sufficient, but source localisation can be used as additional cue according to the principle of multiple mapping.

3.3.3.3 "Cartoonification" or the influence of room information

Room information can be used to increase the impression of being immersed within a "real" sounding environment. In [26], the terminus of *ambience labelling information* is used. It concerns the relationship between sounding objects and their environment and provides context information - for instance synthetic room information. Ambience labelling information can be used to express perceptual significance:

One axis of perceptual space ranges from foreground to background or from *maximum significance* to *completely ignored*, with an infinite number of gradations. To use perceptual significance within auditory user interfaces, the number of gradations must be reduced to several discrete steps or *cartoonified*: An acoustic object which is addressed to be in the perceptual background can be cartoonified by adding room information and distance coding. Additionally, the cartoonification could be expressed by a reduced sound quality (e.g. presenting less significant sources with a reduced resolution with regard to source position and sound quantification) or by applying low-pass filters to create a muffled sound.

Limitations of adding room information

Adding room information to the sound items within an auditory interface is limited by a reduced speech intelligibility and a diminished localisation accuracy, because the temporal envelope of the direct sound signal is smeared and transients are lost. A classification of natural rooms according to their usability for speech performances generally includes the evaluation of the energy distribution vs. time. As an example, one often used measurement for speech intelligibility, C_{50}, is given:

$$C_{50} = 10 \cdot \log \frac{\int_{0ms}^{50ms} h^2(t) \cdot dt}{\int_{50ms}^{\infty} h^2(t) \cdot dt} [dB]$$

For C_{50}, energy is calculated by a backward integration of the squared impulse response h(t). If C_{50} exceeds 2 dB (which means that more than 60% of the total energy lie within the first 50 ms) speech intelligibility in this room is good. If the room contains too much energy after the first 50 ms, speech intelligibility is decreased.

To evaluate the effects of room information on spatial unmasking, the simultaneous masking of concurrent sounds is divided into two categories: *informational* masking and *energetic* masking. Informational masking occurs, when the concurrent sources both are audible (which means there is only minimal spectral overlap) and therefore are easily confused. Energetic masking, for example, is the masking of a speech sound with broad band noise – there definitely is spectral overlap, and the speech sound is less audible. In [38] it is shown, that the improvement caused by spatial unmasking is less affected, when room information is added to the sound sources and only informational masking occurs. In the case of energetic masking, the effect of spatial unmasking is degraded by available room information.

In a 3D virtual environment which enables the user to move between the sound items, the negative effects of adding room information loose their impact: The user perceives a “direct” and intelligible signal of the sources that are close to him, and a distant and reverberant signal of the distant sources.

4. EVALUATION OF THE TESTS

To validate the collection of mapping strategies described in chapter 3 of this thesis and to test the usability of the interaction patterns, two listening tests were conducted. The evaluation of these tests is content of this chapter.

4.1 Test implementation with Ambisonics and pd

For the 3D presentation of the AUI for the test applications, the sound sources must be spatialised within a simulated room. The layout should enable the spatial presentation of several sound sources at the same time and the movement of the user between the sources. The used sound reproduction scheme must fulfil the following requirements:

- The reproduction of the sound sources must include spatial information, room information and distance coding at an appropriate quality to ensure localisation accuracy, dynamic accuracy and externalisation of the sources while remaining intelligible.
- For more flexibility and portability, the system must enable binaural reproduction via headphones as well as reproduction with loudspeakers.
- The given feedback must include translation between the sound sources and head rotation. The system must be able to deal with a dynamically changing layout relative to the position of the user and with a dynamically changing content.
- To evaluate the movements and interactions of the test users, the system must be able to export corresponding data in a suitable temporal resolution.
- The system must remain executable on a customary PC.

As rendering engine for the two listening tests, an Ambisonics [21] encoding and decoding system combined with a room simulation and a binaural encoding stage was used: A real time rendering engine for binaural reproduction of 3D Sound with the use of Ambisonics as intermediate rendering dimension which fulfils the above listed conditions was developed at the Institute of Electronic Music and Acoustics. Detailed information about this engine can be found in [32].

4.1.1 Ambisonics encoding/decoding

The Ambisonics approach is an approximation of a holographic system - the reproduction of the sound pressure inside a source free volume with an infinite number of sources placed at the surface of this volume. An incoming (plane) wave is developed into a series of *spherical harmonics* (4.2), where Θ and Φ represent the dependency from azimuth angle and elevation (4.1), σ is $\pm$ 1, and the indices m and η run from 0 to ∞. The development of the series is broken off at a certain order which determines the order M of the Ambisonics system.

$$p(\vec{r}) \cong \Theta(\vartheta) * \Phi(\varphi) = Y^{\sigma}_{m,\eta}(\vec{r}) \tag{4.1}$$

$$B^{\sigma}_{m,\eta} = Y^{\sigma}_{m,\eta}(\Phi,\Theta) * p_{\Phi,\Theta} \tag{4.2}$$

The thus encoded signal is decoded to a virtual speaker layout, the minimal number of required speakers L is determined by the order M of the Ambisonics system and also depends on the used dimensions (2D or 3D):

$$L^{3D} = (M+1)^2 \tag{4.3}$$

$$L^{2D} = 2M+1 \tag{4.4}$$

The decoding process corresponds to the distribution of the Ambisonics channels over the used loudspeaker layout and is performed by comparing the coefficients of the incoming wave with the summation of plane waves, reproducible by the set of loudspeakers. This leads to the so-called *matching conditions* (4.5), where the left sides of the equations are the description of the incoming plane wave and the right sides are the summation of the N loudspeaker signals.

$$s = \sum_{n=1}^{N} p_n \tag{4.5}$$

$$s * P_{m,\eta} * \cos(\Theta)\cos(\eta\Theta) = \sum_{n=1}^{N} p_n * P_{m,\eta} * \cos(\vartheta_n)\cos(\eta\varphi_n)$$

$$s * P_{m,\eta} * \cos(\Theta)\sin(\eta\Theta) = \sum_{n=1}^{N} p_n * P_{m,\eta} * \cos(\vartheta_n)\sin(\eta\varphi_n)$$

The reproduction of the incoming wave with a limited set of speakers can be regarded as spatial Fourier transformation, and the panning of the Ambisonics signals can be described by the so-called angular sinc functions. Therefore, the panning of a virtual source can cause localisation errors because of out-of-phase signals produced by adjacent speakers. This is attenuated by applying windowing techniques across the different orders within the Ambisonics domain, paid with an increase of the localisation blur.

Rotation of the complete sound filed can be easily achieved within the Ambisonics domain by simple matrix multiplication, therefore the Ambisonics approach is especially suited for virtual environments which enable head movement.

4.1.2 Room simulation

To improve the perceived externality of the virtual sound sources and the impression of realism, a two-staged room simulation is added to the rendering engine [32]:

- Early reflections are calculated by creating image sources due to the rules of geometrical acoustics. The signals from the image sources are delayed, attenuated and low-pass-filtered for a more natural impression.
- Late reverb is added by a circuit of all pass filters building a large global recursive network.

4.1.3 Binaural mix

The binaural mix of the Ambisonics signals is performed with HRTF filtering. For the rendering engine, non-individualised HRTFs (from the KEMAR filter set) are used. The Ambisonics decoder produces a number of output channels, depending on the Ambisonics order (4.3 and 4.4). Each output channel is related with the position of a virtual loudspeaker – the momentary angle between this speaker and the listener determines the required HRTF filter.

This illustrates the advantage of Ambisonics as rendering strategy instead of a direct positioning of each source: Independent of the number of virtual sources within the interface and independent of their movement, the required set of HRTF filters is static. Each output channel causes two filtering operations – one for each ear. Thus, an increased number of sources (or image sources required for the room simulation) only increases the computational load of the Ambisonics encoding stage but leaves the decoding stage and the number of channels for the binaural mix unchanged. As the binaural mix is a highly computational filtering operation, this is of great advantage.

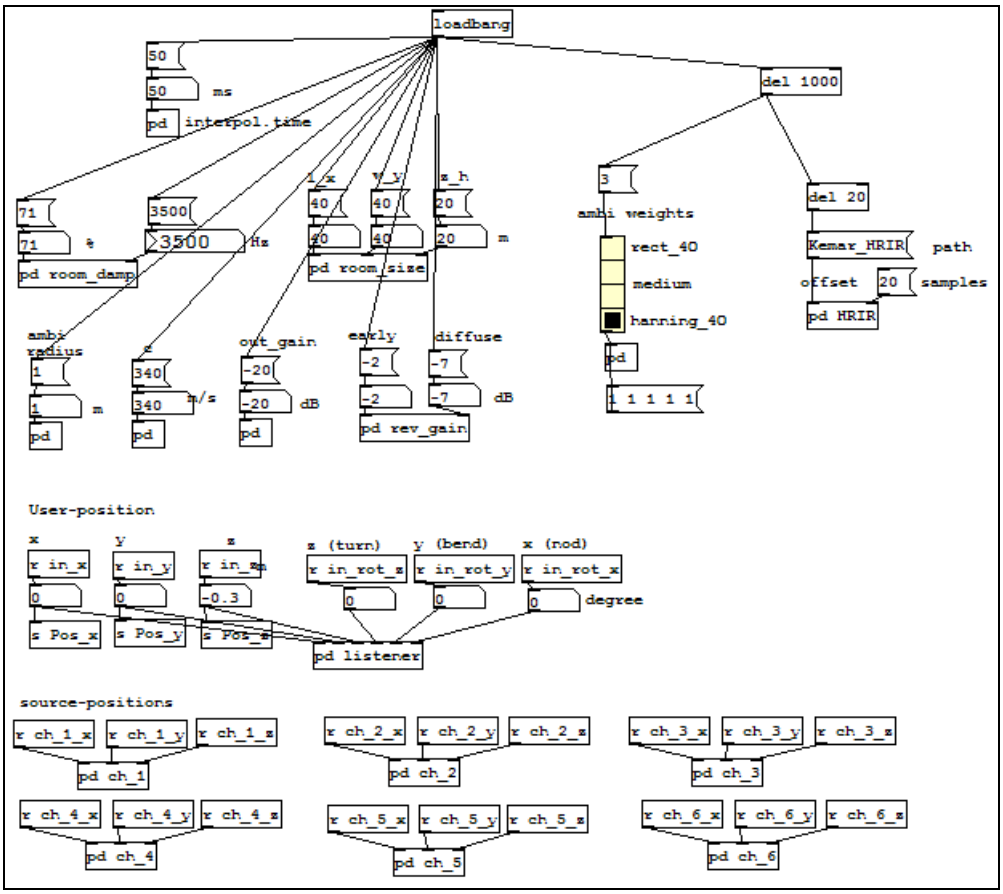
The binaural output is created by summation of all output signals for the left and the right ear.

4.1.4 Implementation with pd

The binaural rendering engine is implemented using Pure Data (PD – a graphically based real time computer music software by Miller Puckette. PD is an open source software.

[Fig 12] shows the part of the rendering engine which enables the modification of the Ambisonics parameters and the room properties. The windows for the weighting function can be varied from a rectangular window to a hanning window. The room size can be entered in meters, and the room properties can be varied by changing the amplitudes of early reflections and late reverb. Furthermore, the frequency range for the low-pass filtering of the early reflections and the late reverb can be modified.

The absolute position of the user within the virtual room can be entered in x, y and z co-ordinates, and the angular head position of the user, e.g. collected by a head-tracking system, can be entered as rotation around X-,Y- and Z-axis. For both test applications, a maximum of six sources at the same time with variable source positions was required.



[Fig 12] Screenshot of the rendering engine showing the Ambisonics control parameters

4.2 Evaluation of test 1 – The Shop

4.2.1 Test purpose

The purpose of the first test was to investigate, whether a simple graphical application can be transferred into the auditory domain by simulating the 2D screen layout within a 3D auditory environment. To address the different properties of the visual and the auditory system, an acoustical lens was implemented to mimic the visual ability to focus on one object or to look at the complete screen. Interaction was possible via head rotation and a conventional keyboard. The application was encoded with Auditory Icons and naturally recorded speech, and the mapping procedure was performed without the aid of interaction patterns.

4.2.2 Test application

The test application was a shopping-program for goods of daily life which included typical elements of a user interface like a structured menu, exclusive and nonexclusive selection from a list, fields for text entries and confirmations of selections and notifications.

Title Menus:	**File**	**Goods**	**Order**
Submenus:	New List	Fruit	Delivery
	Search	Bread	Payment
	Close	Milk products	
		Beverages	

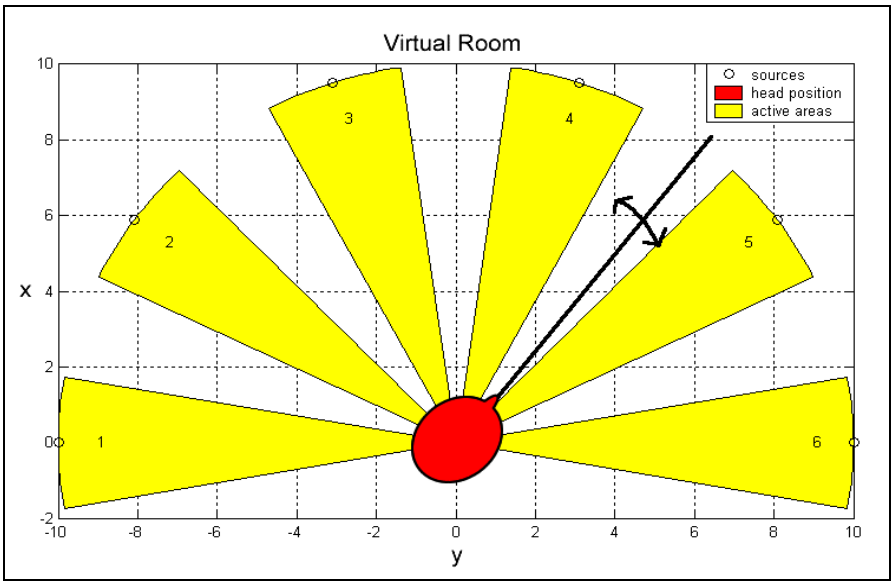
[Tab 1] Menu Structure of the first test application

The menu is structured in three title menus with different sub-menus. Each submenu leads to a corresponding dialog, where different actions can be performed by the user. The required actions to fulfil the tasks were:

- Answering different Yes/No dialogues or alerts
- Entering data (e.g. personal data) to dialogue windows
- Performing exclusive selections from lists with two options
- Performing non-exclusive selections from lists with four options

4.2.3 Virtual room, source positions and focus

All items that would occur in a corresponding graphical user application were represented through Auditory Icons. These icons were placed in a virtual room which contained a maximum of six Auditory Icons at the same time. The Icons were positioned on a semicircle at six fixed positions in a range from –90° to +90° out of the median plane [Fig 13]. In the virtual centre of this semicircle, the user had a fixed position, and the only possible movement was head rotation. Although presented within a 3D environment, the layout of the interface was two-dimensional, and the possible movement was in fact one-dimensional. Each sub-menu or dialogue was situated in a separate virtual room. To leave this room, interaction with one of the Auditory Icons was required.

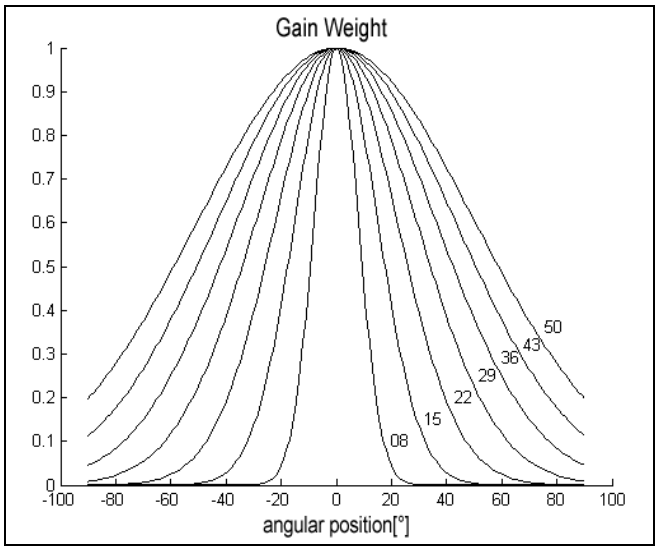


[Fig 13] Source positions and active areas within the virtual room

To mimic the visual ability of focusing on a certain area of the monitor, an acoustic zoom function was implemented. This was done by weighting the input gain of the sources and not by weighting the Ambisonics channels, giving the advantage of an unlimited steep weighting function without negative effects in the frequency domain. The weighting function did not affect the early reflections and the reverb, so the room size did not shrink with the use of the zoom function. The zooming function was implemented with a Gaussian window (4.6). X is the angle from –90° to +90°, m(t) is the momentary head position in degrees and s is the steepness-parameter of the window, ranging from 8 to 50.

$$w(x) = e^{\frac{(x-m(t))^2}{-2*s^2}} \tag{4.6}$$

[Fig 14] shows the zoom function for a head position of 0°, plotted with the zoom factor as parameter. The zoom factor varies from the minimal factor of 8 (only one source can be heard at once) to the maximal factor of 50 (all sources can be heard concurrently, but there is a stronger weight on the sources in front of the user).



[Fig 14] Gaussian zoom function

4.2.4 Interaction with the interface

Interaction with the application was performed by the head position of the participant, which was recorded with a magnetic headtracker mounted on headphones, and by a conventional keyboard. The virtual room was divided into several active areas (a region of +/-10° around the position of each icon [Fig 13]). If the user turned his head towards one of the active areas, the contained icon became active: Any input via keyboard applied to this icon only. The functionality of the keyboard was as follows:

- The spacebar was used for clicking.
- The F1 and F2 buttons were used for acoustical helping functions: The F1 button informed about the active sources, the F2 button gave a hint about the submenu or dialogue, in which the user currently moved.
- With the arrow-buttons, the zoom function was adjusted. The “up”-button increased the zoom factor (= zooming out), the “down”-button decreased it (= zooming in). By

default, the zoom factor was in the mid-range. If a new dialog or submenu was opened, the zoom factor was reset to a value in the mid range.

- The Esc-button led back to the main menu. It worked within the submenus only. The dialogs had to be left with “OK” or “Cancel”-buttons.

4.2.5 Auditory Icons, iconic language and pedestal tones

For most Auditory Icons, sounds were chosen intuitively fitting to the represented object, because the participants had only a limited amount of time to learn the meaning of several different icons. For the items within the selective dialogues, no specific icons were created, because Auditory Icons for similar data (e.g. different types of milk products) would hardly be distinguishable. The options within these dialogues were expressed by short piano- and violin-samples of different pitch. To find out which item lay hidden behind each option, the user had to press the F1-button.

For some of the general sounds, an *iconic language* was created. A description of this concept can be found in chapter 3.3.1.2, page 27 of this thesis. Iconic language was used for buttons (OK, Cancel) and to express the status of the selectable list elements.

All Auditory Icons were played in a loop. As they are recordings of natural sounds and their gain envelope is fluctuating, localisation can be difficult. To overcome this, *pedestal tones* were created: A continuos musical tone was added to each source position with increasing pitch from the left to the right (f_{ground} = 392, 523, 659, 784, 932, 1046 [Hz] or g^1, c^2, e^2, g^2, b^2, and c^3). The gain was kept lower than the gain of the icons. Each musical tone was a combination of four harmonic partial tones. Additionally, a slight amplitude vibrato was added to make the tones easier to distinguish.

4.2.6 Test users and test procedure

The test was performed by a group of ten test participants. Among them were four participants who were blind or visually impaired. After an instruction, the participants got 15 minutes of training time with the application, and then they had different tasks to fulfil. (see Appendix III). The tasks were explained verbally to have equal conditions for people with normal and impaired visual abilities throughout the test. The movement within the virtual room and all performed actions were recorded. The test was supervised and video-taped, and after the test, all participants were asked questions about their subjective impression.

4.2.7 Test results

In this section, a short review of the results of the first test is given. A more detailed evaluation and several graphs can be found in Appendix III of the thesis.

Usability of the virtual layout

For the purpose of this test, the hemispherical, in fact one-dimensional test layout worked well. The total number of sources played concurrently was always limited to a maximum of six sources, for which the available range was sufficient. An increase of the maximum number to eight or ten concurrent sources would already be critical with regard to the limited angular range of human head rotation and the relatively rough resolution of localisation which can be achieved with it. For real-world applications, the limitation to six or eight elements presented at one time is far too small - even if only parts of the application are represented at the same time. Therefore, an alternative layout has to be found, ideally one which allows movement into more dimensions and which can host more sources than the circular layout of the first test.

Usability of the zooming function

In the test application, all sound sources were played concurrently. If the user entered a new dialogue, he was at once confronted with a burst of six Auditory Icons, and he had to filter the useful information, before any interaction could take place. For this purpose, the user needed either very trained ears or the aid of the zooming function.

The first test showed, that the localisation abilities of the test users were very different and that many of them had to rely completely on the zooming-function and adjusted it to maximum zoom most of the time. This annoying procedure of zooming before receiving the required information can not be used for real-world applications. Even throughout the test, some of the participants completely stopped zooming and repeatedly pressed the helping function, until the searched item was found. This clearly indicates, that a different approach for concurrent or successive temporal placement of the Auditory Icons has to be found.

Usability of the interaction process

The test has shown, that the used interaction process as combination of head rotation and keyboard input is difficult to handle for blind and sighted users as well. To select an item, the user had to locate the desired item via head rotation. Then he had to maintain this position until confirming the selection via keyboard. A small shift in the head position was enough to lose the item, and it had to be relocated again. For future applications, an alternative input device which can be utilised by blind and sighted people must be used.

Usability of the Auditory Icons

All items within the application were represented by Auditory Icons. Apart from the fact, that the concurrent representation of six sounds, played in an endless loop, is stressful for the user, the sound design of the Auditory Icons was not a very pleasant one. Only four of ten participants found none of the icons annoying.

For future applications, there has to be a "quiet zone" without any noise somewhere within the virtual room, and the usage of Auditory Icons has to be reconsidered – to convey information verbally would be far less annoying for blind users and it would be far more versatile: The test users were able to learn a limited set of icons within a short amount of time, but throughout the test, they had to consult the helping function several times. The opinions, whether the icons were intuitive or not, differed.

Usability of the test application as a whole

If the test performances of blind and normal sighted users are compared, there is only a slight difference between the two groups (see Appendix III for details), which is encouraging. If the shortcomings with regard to interaction and sound-design can be reduced within the second test, a prove of the ability of spatial AUIs to represent complicated graphical applications can be given.

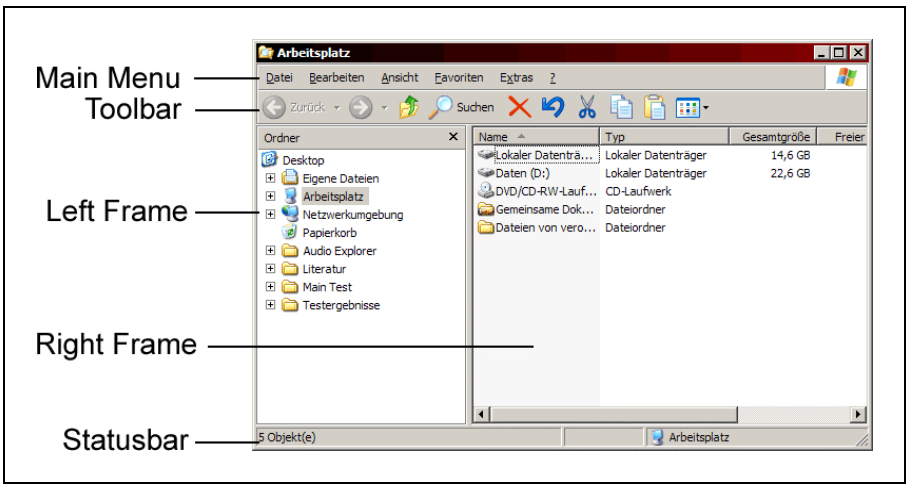
4.3 Evaluation of test 2 – The Explorer

4.3.1 Test purpose

Based upon the results of the first test, a second test was performed. In contrast to the first test, where a simple test application was especially invented for testing purposes, for this test a more complex real-world-application which could be expected to be known to all test users was transferred into the auditory domain. As test application, the MS Data Explorer was chosen. For the transferring process, interaction patterns were used. To address the shortcomings of the first test layout, within the second test, different strategies to convey information, a different sound design and a different layout of the virtual room were used.

4.3.2 Test application

For the implementation of the test application, the available functions of the MS Explorer were listed and, for the testing purpose, the complexity of the application was reduced. Some of the rather complicated functions which cannot be assumed to be familiar to all test users were crossed, and the basic functions remained. This remaining content can be divided into several major parts, illustrated by [Fig 15].



[Fig 15] Major parts of MS Explorer

The following section gives an overview of the parts which were implemented for the test application.

- *Main menu*: The “Extras” menu header and some menu items within the sub-menus were crossed completely. The following menu structure remained:

File	**Edit**	**View**	**Favourites**	**?**
New Folder	Undo	Statusbar	www.iem.at	Information
Delete	Cut	List	www.orf.at	Support
Rename	Copy	Details	www.google.com	
Properties	Paste	Sort		
Close	Select All	Change To		
		Reload		

[Tab 2] Menu structure of the second test application

- From the *standard buttons toolbar*, five buttons were implemented: Undo, Delete, Cut, Copy, Paste.

- *Left and right frame*: The left frame shows the hierarchy of folders in a tree structure representation with the desktop as root. The right frame shows the content of the folder which is currently selected within the left frame. The view options “list” and “details” are possible. The selections or modifications within the left frame determine the content of the right frame: It always displays the content of the folder which is currently selected within the left frame.

- The following *context menu* (right click menu) was implemented:

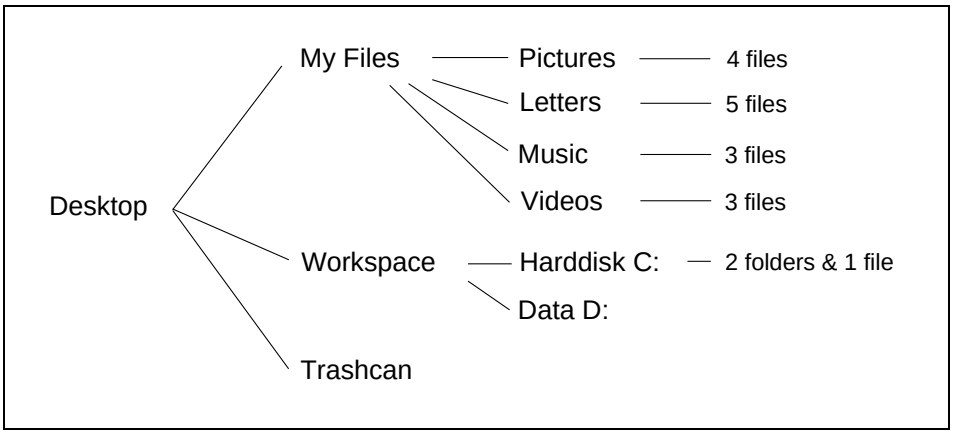
Cut
Copy
Delete
Rename
Properties

[Tab 3] Implemented context menu

- *Emerging Windows*: Some of the menu items lead to separate windows, where information is presented or modifications can be performed. From this windows, only the “properties”–window was implemented. Additionally, a second window was implemented which appears, if a file or folder is deleted. This window contains the question, whether the file or folder shall be deleted or not, and gives the possibility to answer with OK or Cancel.

4.3.3 Structure of files and folders

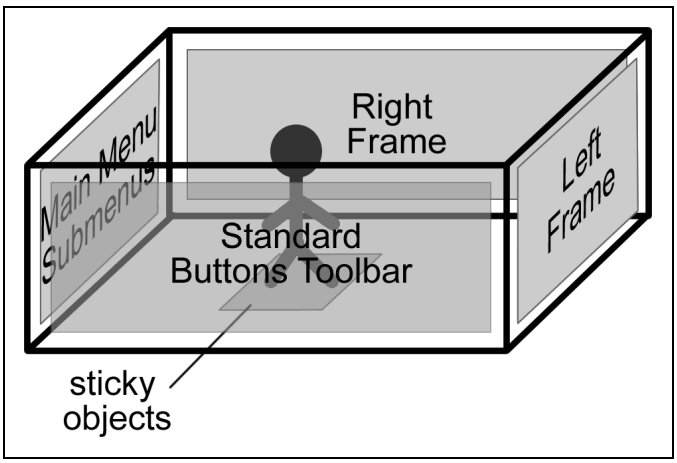
For the test application, a fictitious structure of 16 files and 12 folders with the desktop as root and 4 different hierarchic levels was invented. A structural overview of the used files and folders is given in [Fig 16].



[Fig 16] Used structure of files and folders

4.3.4 Virtual room and source positions

All items that occur in the graphical counterpart of the test application were represented through sounding items, which were placed on the borders of a 3D virtual room. Each wall hosted a different part of the MS Explorer. The front side wall and the right side wall were reserved for the right and the left frame of the main window. The menu structure was placed on the left side wall, and the standard buttons toolbar could be found on the rear wall. Important objects which required immediate user attention were implemented as sticky objects: They moved with the user and therefore were always placed directly in front of him [Fig 17].

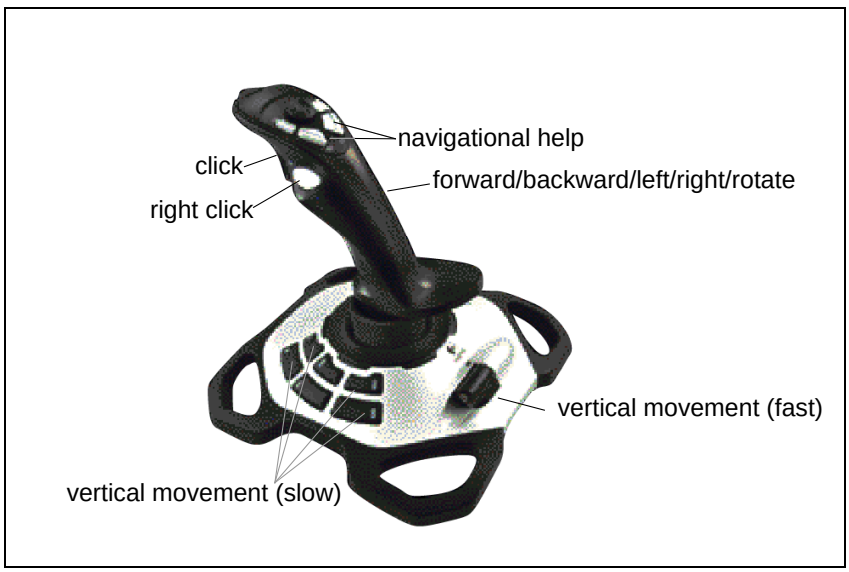


[Fig 17] Mapping of the MS Explorer onto four walls of the virtual room

The sound items were not played concurrently, but were triggered by the movement of the joystick: A grid with ten lines in x and y direction was projected onto the ground plane of the virtual room, producing a grid of 100 cells. In the midst of each cell, an auditory object was placed. If the user moved the joystick across this cell, the corresponding sound item was triggered and played by the interface. Despite of their triggered reproduction, the sound sources were subject to room information and distance coding: Once triggered, the user had a full room and distance simulation of the source.

4.3.5 Interaction with the interface

The joystick was the only exploited input device [Fig 18]. It was used for movements on the ground plane (including forward/backward and left/right movement and rotation around the vertical axis) and for vertical movements. For clicking and right-clicking, two of the buttons on the front side on the handle were used. Two further buttons on the top of the handle controlled a helping function that eased navigation within the virtual room. A fast vertical movement was possible with the thrust control of the joystick. A slower vertical movement could be performed with the use of the buttons on the socket of the joystick. To ease navigation, the audio sources were placed alongside the walls of the virtual room which were mapped to the borders of the possible joystick movement. With this restriction, all sources could be located by the user quickly.



[Fig 18] Joystick with input possibilities

4.3.6 The sound items

For the second test, the majority of sound objects were coded with speech samples (recorded of different natural speakers), but also Auditory Icons and Continuous Sound were used. The transformation process was completely performed with the aid of interaction patterns:

For the Menu headers and the standard buttons toolbar, the *Command Area* pattern was used. Consistent to the menu headers, the sub menus were transformed by using the *Contextual Menu* pattern. Menu items (menu headers and sub menu items) were represented as combination of recorded speech and an instrumental tone. The menu headers were placed alongside the left wall. Moving from the first item to the last one, the instrumental tone added to the spoken name of the menu item increased in pitch. If the user clicked on one of the menu headers, the sub-menu opened vertically. The desired sub-menu item was selectable by moving upwards and clicking on it. Again, with an upward movement, the pitch of the instrumental sound increased. For the standard buttons toolbar, each button or *Triggering Element* atom was represented as a combination of an Auditory Icon and, delayed, a spoken hint. The related function was started with a click.

For the structure of the left and the right frame, the *Container Navigation* pattern was used. In the left frame, the hierarchic structure of folders was represented by the *Tree Structure* atom. Each folder within the hierarchy was spoken by a male voice. The hierarchy started at the bottom of the virtual room with the desktop. A movement upwards brought the user to further folders (My Files, Workspace…). For parent folders, additionally a high pitched „pling"-sound was played (corresponding the "+"-sign in the graphical tree structure). If the parent folder was selected with a click, the "pling"-sound was replaced by a lower-pitched version of itself. The hierarchic structure within the first region was represented by different spatial positions of the sources: The leftmost source had the highest hierarchic position, the rightmost source the lowest one.

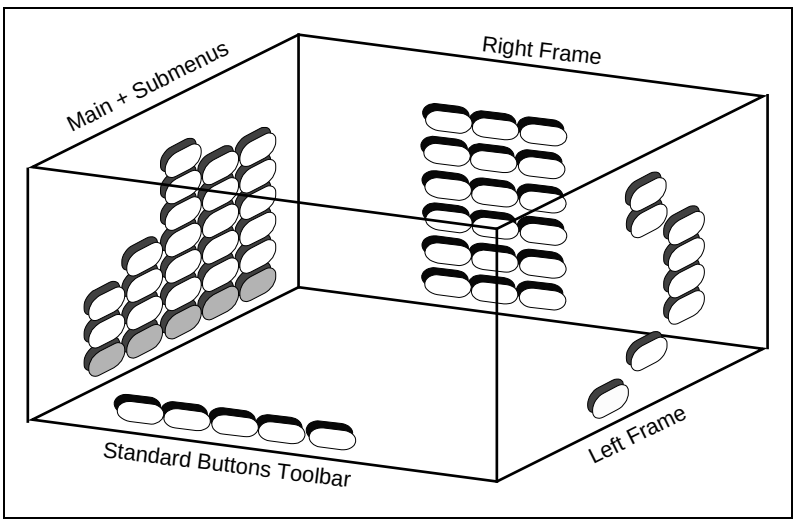
For the content of the right frame, the *List* atom was required. Files and folders were distinguished by the voice of the speaker: Files were spoken by a female voice, folders were spoken by a male voice. If the representation mode "list" was selected within the main menu, only the names of the files and folders were spoken. If "details" was active, file properties were added after the name. Again, element selection was performed with a single click.

To open a folder within the second region, double click was required. The sorting mechanism was not implemented for the test application.

For the right-click or context menu, again the *Contextual Menu* pattern was used. Like in the graphical application, it was triggered by right-clicking on a file or folder and was opened vertically. The sounds and interaction processes of the context menu were chosen consistently to the menu structure on the left wall.

For the Windows, the *Message* pattern was used. Windows were distinguished from the normal environment by background music. When the window opened, the content message was spoken. This message could be repeated by a downwards movement with the joystick. To explore further content (details, buttons…), the user had to move upwards. For the buttons, Auditory Icons were used. The feedback of the pressed button was given by a clicking sound. The same concept was also used for the appearing alerts. Windows and alerts were implemented as sticky objects and always remained directly in front of the user.

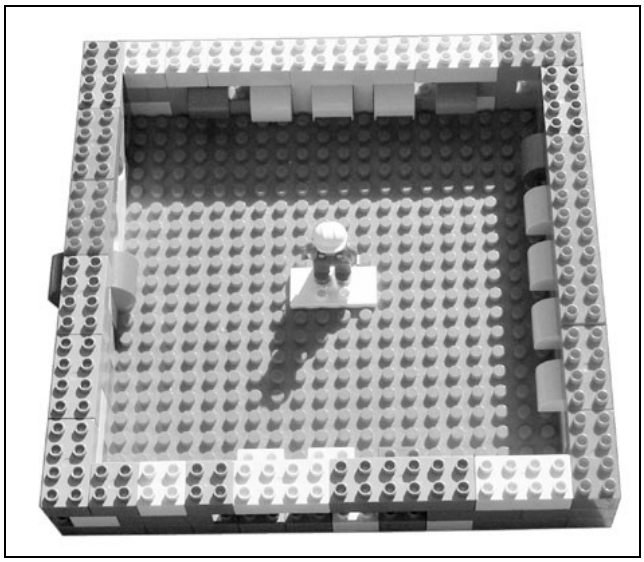
To ease orientation within the room, a navigational help function was implemented: Upon buttonpress, a rough information about the current position of the user was given. The final spatial layout of all sources within the virtual room is shown in [Fig 19].



[Fig 19] Distribution of audio sources within the virtual room

4.3.7 Test users and test procedure

The test was performed by a group of 15 test participants. Among them were seven participants who were blind or visually impaired. After an instruction, the participants got 15 minutes of training time with the application, and then a list of tasks to perform was given (Appendix III). The tasks were explained verbally to have equal conditions for people with normal and impaired visual abilities throughout the test. Additionally, a miniature room model was built, which was presented to all users [Fig 20]. Again, movements and interactions with the interface were recorded, and subjective questions were asked after the test.



[Fig 20] Miniature model of the test application

4.3.8 Test results

In this section, a short review of the results of the first test is given. A more detailed evaluation and several graphs can be found in Appendix III of the thesis.

Usability of the virtual layout

The three-dimensional layout of the virtual room with the different meanings of the four surrounding walls and the two-staged movement (ground-plane movement towards the walls, vertical movement for selection) proved to be sufficient to host the elements of a real-world application. On the ground floor, at least 20 items (5 on each wall) can be placed, not to mention the potential with regard to vertical placement.

The thematic grouping of elements on the different walls was easy to memorise for the test users. The usability of the mappings of particular interaction patterns is different. While the menu structure was easy to use for most test participants, the representation of the folder hierarchy is in need for improvement: Hardly any test user had a clear overview of the file and folder structure. The mapping of the folder hierarchy onto the spatial location of the sound item is unusable for this purpose.

Usability of triggered sound reproduction

The quasi-sequential sound reproduction triggered by the movement of the joystick proved to be a good strategy for placing several sound sources within one virtual room. The relation between joystick position and triggering mechanism was obvious for the test users, and the spatial impression of the virtual room was enhanced. Hearing mainly one source at a time was also far less stressful for the test users than a concurrent reproduction. The room impression was maintained by the fact, that the triggered sources still were subject to room simulation and distance coding.

With the triggered sound reproduction, the multiple representation of information within the interface was in fact abandoned. Still, experienced users who quickly moved through the virtual room triggered the sources very fast and caused the concurrent replay of up to three or four sources. Thus, the triggered sound reproduction is a compromise between parallel presentation of information while avoiding too much cognitive load and annoyance. Furthermore, it supports usage by novice users and experts as well by an individually regulated working speed. The subjective impression of the test administrator with regard to user acceptance was, compared to the first test, a very positive one. The participants seemed to enjoy working with the application and were ambitious to fulfil all of the given tasks.

Usability of the interaction process

The joystick input mechanism proved to be usable for blind users as well as for sighted users, although the available feedback given through the handle of the joystick may be more helpful for the sighted ones. The blind users tended to move in regions without any active areas more often than their sighted colleagues and had to search slightly longer for the sources.

They certainly had difficulties with the distinction between clicking, double-clicking and right-clicking, because with regard to this aspect the sighted users had more knowledge in advance. Also the input procedure for vertical movement is in need for improvement.

Nevertheless, the test has shown, that a joystick as pointing device is usable and accepted by blind users. With the joystick, the participants also had the ability to control their working speed with the velocity of the movement.

Usability of the sounds

The answers to the subjective questionnaire show, that the test users distinguished between the different items within the application (menu item/file/folder) according to the different speakers *and* the different spatial positions to equal parts – multiple mapping and expressing differences by changing more than just one sound attribute supported the functionality of the application. Only one user was annoyed by the sounds, namely by the Auditory Icons used for the standard buttons toolbar. Obviously, the usage of speech representation instead of auditory icons proved to be a good idea.

Usability of the test application as a whole

One goal of the application was to reach equal usability for blind and sighted users. With regard to the main performance measurements like the total test time, the required selective operations and menu handling operations and the working speed, both groups reached similar results with a slight advantage for the sighted group, although there is a high standard deviation for some values (Appendix III).

With regard to complexity, the second test application exceeded the layout from the first test by far [Tab 4]. Still, the subjective ratings given by the test users after the tests reach better results for the more complex second application – the changed strategies and the usage of the interaction patterns are responsible for this improvement.

Item	Test 1	Test 2
Main menus	3	5
Submenus, total no.	9	21
Additional content	6 Lists with a total number of 20 selectable items 4 Text fields	File structure with 4 hierarchic levels and 1/3/6/18 elements respectively Context menu with 5 items
Buttons	2 different buttons	7 different buttons

[Tab 4] Comparison of the two test applications with regard to complexity

5. CONCLUSIONS

5.1 Usability of Auditory User Interfaces

The main goal of this thesis was to investigate, whether a rather complex real-world application can be transformed into the auditory domain in a way which can provide the benefits from a GUI also to blind and visually impaired users within the auditory representation.

The second listening test has shown, that the transformation of an application including dialogues, menu structures, toolbars and hierarchic structures into the auditory domain is possible, and that three-dimensional navigation and interaction within the interface is feasible for both blind and sighted users being equipped with *the same hard- and software*. Both groups of test users were able to memorise the spatial concept which lay behind the test application within a short amount of time.

The spatialisation of sound, the room information and the given feedback of the movement (acoustical feedback of the interface and mechanical feedback given by the joystick) were helpful in this point. Furthermore, the second test caused mainly positive reactions by the blind users. Most of them found the approach interesting and could imagine working with it.

5.2 Evaluation of mapping strategies

As the initial examples in chapter 1 have shown, the usability of auditory applications varies strongly and depends on the chosen layout. For instance, for a rather simple application like a time scheduling application [43] a fitting design within the auditory domain was found, and the acoustic version had equivalent or even improved usability compared to the graphical counterpart. To gain the same result for rather complicated applications or even for a set of arbitrary applications is more challenging.

A comparison between the two listening tests with regard to the used mapping strategies clearly favours the second strategy: The concurrent presentation of sounds was limited to a maximum of two or three sounds at the same time, triggered by the movement, and in a central position, a *silent spot* was offered. Furthermore the usage of Auditory Icons and Earcons was reduced, compared to the first test, and, whenever possible, replaced by short samples of recorded speech. Although the second test layout was more complex and included much more elements than the first test, both training time and subjective ratings of the test users brought better results for it. After all, the exchange of information with spoken words is the basis of human communication, and especially blind users are very skilled to attending a spoken representation of their computational work.

Nevertheless, also the design of the second test included several shortcomings which will have to be addressed for future work. For instance, the representation of a hierarchical tree structure by recorded speech for the names of files and folders and by spatial position caused major problems for both blind and sighted test users. Orientation was difficult within the tree structure and the test participants had problems to recognise in which directory they were currently working. Other parts like the main menu or the standard buttons toolbar hardly caused any problems. This brings forth an advantage of the used design strategy: The idea to decode the content of an application according to its ability to solve user interaction problems proved to be a good one, because it enables a modular mapping of the independent interaction patterns and with it a stepwise movement towards a “good” design for the whole application.

The second listening test has also shown, that the assignment of usability problems to specific interaction patterns enables a user-centred design which is a prerequisite of usability. With the aid of the generalised interaction patterns, the creation of a usable application within a completely new mode of representation was possible.

Furthermore, this design strategy not only enables the encoding of one specific application. Once a usable layout is found, it can be used as model for the transformation of many applications.

5.3 Future Work

Although the results for the second test are encouraging, there is still much research work to do before the usage of auditory interfaces can be considered and their usability can be evaluated in large-scaled listening tests. Some important hindrances have already been taken: The two listening tests have shown, that technical solutions to render the applications acoustically were able to do this in a sufficient quality, and that, if an appropriate design is chosen, the acoustical version of an application is usable and accepted by blind users. The next steps towards a real world application realised within the auditory domain include:

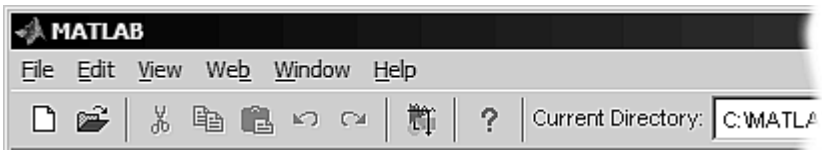
The manual transformation process with the aid of semantic information given by the generalised interaction patterns turned out to be a good strategy in the second test. To enable a multimodal representation of arbitrary applications without manual preparation, a generic depiction process, as proposed in [17], is required. For this purpose, the treatment of automatically extracted information must be investigated. Ideally, also for this purpose a collection of adapted interaction patterns can be defined and automatically recognised.

To exploit common user knowledge, powerful metaphors comparable to the established metaphors of the graphical domain have to be found. This task is related to the vast field of sonification of data and to the psychology of human hearing and must include extensive listening experiments. With regard to this task, also the differences between blind and sighted users have to be addressed.

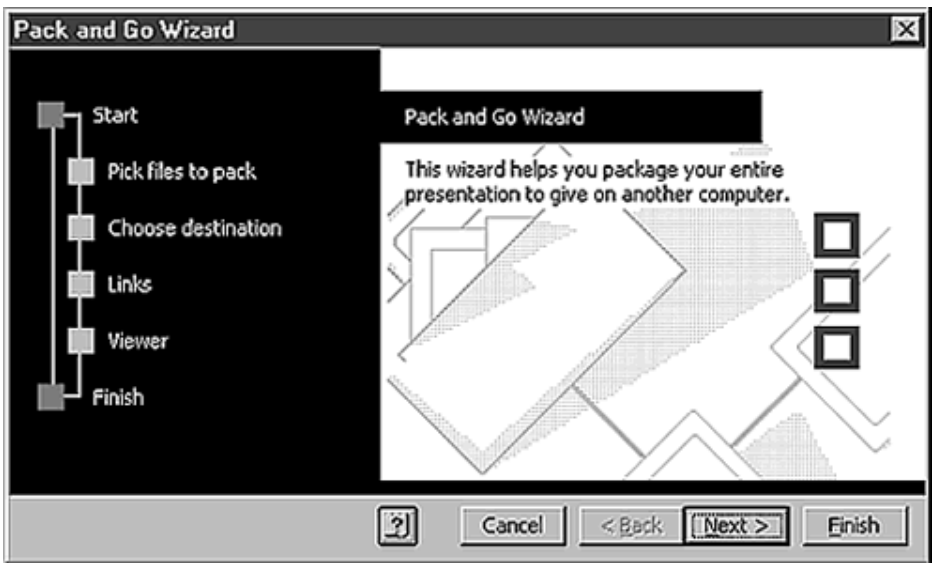
Once more knowledge about acoustical metaphors and their entailments is available, the detailed and thorough mapping of frequently used interaction patterns can be started. If a basic set of mapping rules for these patterns exists, almost any application can be represented acoustically.

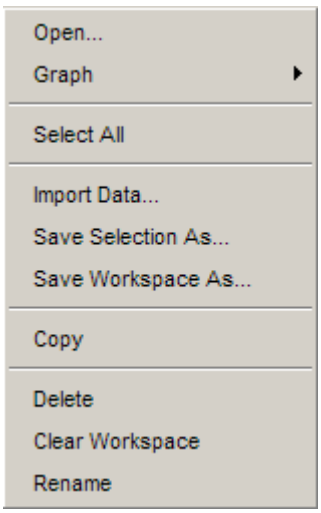
Appendix I: Generalised interaction patterns

In this section, the set of 20 generalised interaction patterns which is proposed after van Welie and Traetteberg [42], is presented. Note, that the patterns have been uncoupled from their graphical approach and generalised. If this generalisation was not required, parts of the original interaction patterns remained unchanged. The wording of these parts may be identical to their original description.

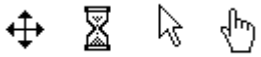
Command Area	
Problem	**The user needs to know where to find the possible commands and how to activate them.**
Conditions	• Immediate access to all available functions increases interaction speed but consumes a large amount of the available representation area. • The main working area should be kept as large as possible. • Concurrent representation of many objects increases the cognitive load. • Some functions are used more often then other functions. • Some functions need additional parameters to be set by the user before they can be executed.
Solutions	**Put the shortcuts to the possible commands in a specific recognisable area** A part of the representation area is reserved for shortcuts to the functions of the application. This area should be distinguishable from other working areas because of contextual attributes. If the number of shortcuts is large, they should be conceptually grouped. The command area can be subdivided to give access to a group of commands. The command area and the included shortcuts should be accessible as direct as possible, especially frequently used shortcuts. Feedback information about the availability of the shortcuts can be given. The number of represented functions should not be too large to limit the necessary representational area and the cognitive load.
Attributes	• Atoms for the representation of the shortcuts (Links, Triggering Elements, Selections). • Separation of groups, when conceptual grouping is required
Graphical Example: Command Area of MATLAB 6.5[1]	

[1] The MathWorks Inc.

Wizard	
Problem	**The user wants to achieve a single goal but several decisions need to be made before the goal can be achieved completely, which may not be known by the user.**
Conditions	• The user may not be familiar or interested in the necessary steps that need to be performed. • The subtasks may not be independent of each other but may depend on a certain sequence. • Several steps are necessary, the required steps may depend on decisions made in previous steps
Solutions	**Take the user through the entire task one step at the time. Let the user step through the tasks and show which steps exist and which have been completed** When the complex task is started, the user is only informed about the goal that will be achieved. To reach the next task, a navigation mechanism (e.g. forward /backward) must be provided. If the completion of one task is obligatory, feedback information must be given (e.g. status information of the navigation mechanism). By navigating back to the previous task, decisions can be revised. At any point in the sequence it is possible to abort the task. Shortcuts can be used to complete all steps in one action. The user needs feedback about his position within the task sequence and about the steps that are part of the sequence. If the whole sequence is completed, feedback has to be given. Each step of the sequence can host interaction patterns.
Attributes	• Atoms (Triggering Elements) for the navigation mechanism • Status information about the navigation mechanism • Atom (Triggering Element) for the exit mechanism • Feedback about the current position within sequence • Information about all steps of the sequence • Feedback when all steps are completed
Graphical Example: Pack and Go Wizard, Example taken from [42].	

Contextual Menu	
Problem	**At any point in time, users need to know what their possibilities are in order to decide what to do.**
Conditions	• The user may not be familiar with the meaning of each possibility. • The number of possibilities may be large and the user needs to locate the desired one • Not all possibilities may be available in the current context, but the user may want to know about all existing options • Integrating learning and using while not hindering the expert
Solutions	**Put the choices in a menu** All available functions are listed in a menu. The list is distinguished from surrounding objects by contextual attributes of the list elements. The functions should be accessible in a single action. If the number of elements is large, they should be sorted to distinguishable groups. To enter this groups, another action may be necessary. For non-expert users, there should be a description available that explains the function of the elements. Functions should be ordered according to one or more of the following criteria: semantics, similarity, frequency of usage or alphabetically.
Attributes	• Atoms (Links) for the representation of list elements • Separation into groups, if necessary
Graphical Example: Right-click menu from MATLAB 6.5[1]	

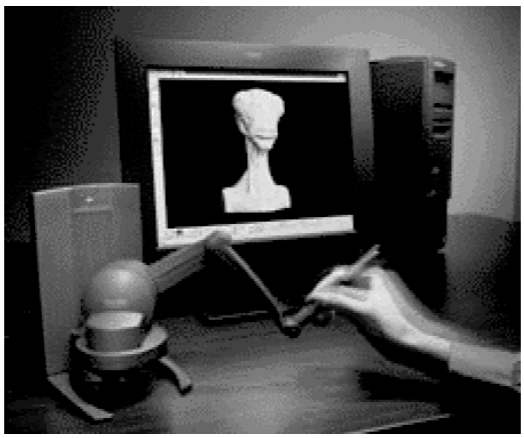
[1] The MathWorks Inc.

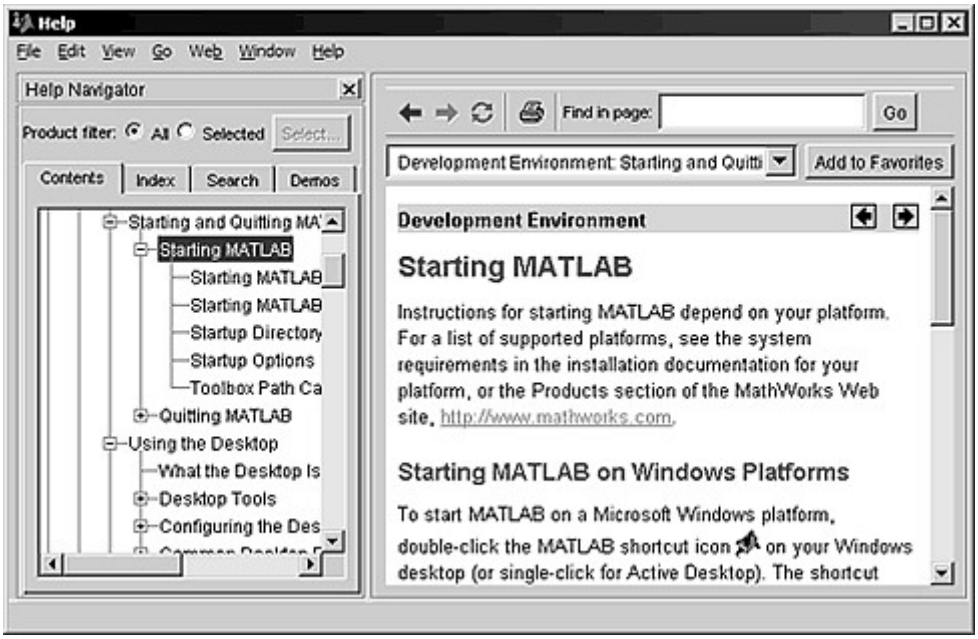
Mode Cursor	
Problem	**The user is creating or modifying an object and needs to know which edit function is selected.**
Conditions	• Usually, objects are modified by selecting the object and applying the desired function. This is slow and unsatisfying when the user wants to apply the same function to several objects. • In graphical programmes functions for object creation are often connected with the cursor. • The user needs immediate feedback on which function was selected, i.e. which mode/state the system is in
Solutions	**Connect the cursor with functions and show the interface state in the cursor** If the cursor is connected with functions for creation and modification, its appearance must change. The cursor can change with regard to several attributes (different shape…). Another possibility is to change the context (e.g. adding the label “delete mode”….). If the function is completed or deactivated, change the cursor back to a neutral cursor.
Attributes	• Different representation for different modes • Type of change (cursor or context)
Graphical Example: Frequently used cursor icons within Windows XP[1].	

[1] Microsoft ® Windows, The Microsoft Corporation

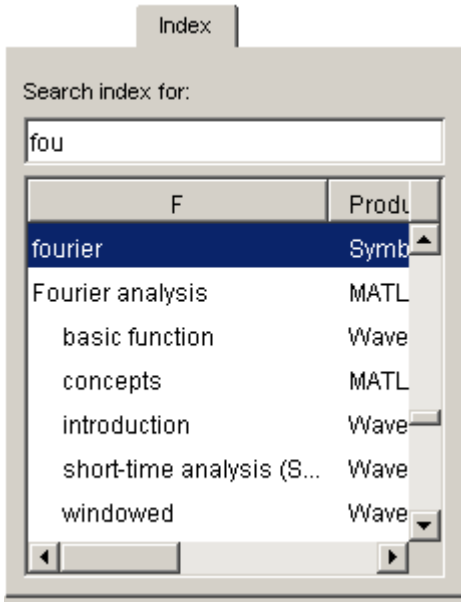
Setting Attributes	
Problem	**Users want to see the attributes of the objects they are working on and additionally they need to know how to modify them.**
Conditions	• Some attributes are partly represented within the document but are difficult to modify by direct manipulation of the objects in the context of the document • Setting attributes requires different controls than representing them • Objects are usually of several types with different sets of attributes • Some attributes are unique for one type of objects, others might be shared by different object types
Solutions	**Create special feedback shortcuts that present the attribute values and can be used to set the attribute values** Parts of the representation area are reserved for shortcuts which can be used to set the attributes of the selected object quickly. The same shortcuts give feedback about the current value of the attributes, always with reference to the selected object. They change their status when another object with different attributes is selected. Only the most common/generic attributes shall be included with the possibility for user customisation. In the case of multi-selection, only the common attribute values are presented. If an attribute of a selected object is not supported, the shortcut must be dimmed.
Attributes	• Atoms for representation of the shortcuts (Links, Triggering Elements) • Representation of attribute values
Graphical Example: Text Attributes in MS Word 2000[1]	

[1] Microsoft ® Word, The Microsoft Corporation

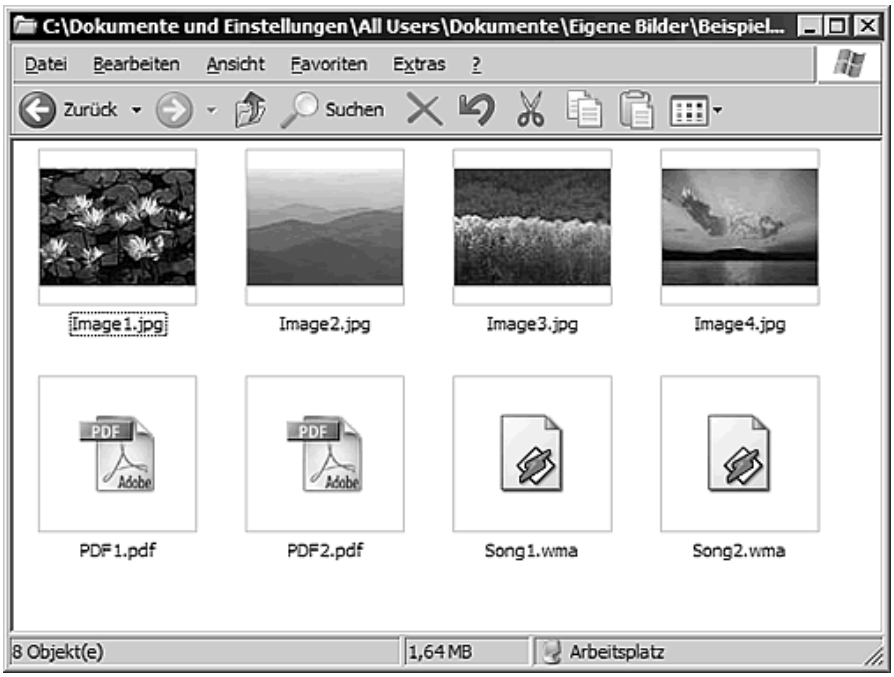
Like in the real world	
Problem	**The user needs to know how to control an object in the interface which resembles an object the user knows from the real world.**
Conditions	• Input devices have a limited set of manipulation possibilities • The way the object is manipulated in real life creates expectations about how the interaction will take place
Solutions	**Match the input device and the widget used** Use a combination of input device and widget that match in terms of possible movements. Movements include the degrees of freedom and types of feedback. For example, if the widget requires rotation around the z-axis, use an input device that supports it. If there is a mismatch choose a different input device or change the widget. From the widget representation it should be obvious, which of the available input devices is required.
Attributes	• General guideline, no attributes
Graphical Example: Interaction device called "The Phantom", used for 3D modelling applications. Example taken from [42].	

List Browser	
Problem	**The user needs to browse or process several items out of a set or list**
Conditions	• The user wants to see an overview of the set/list and at least one item, the representational area might not be sufficient to show both • The user needs to be able to select an individual item as well as to process several items
Solutions	**Find a natural ordering and allow the user to navigate directly from one item to the next and back** The ordering of the list can be done by any kind of ranking scheme. From each item that is presented to the user, navigation to its predecessor and its successor should be possible. If the ranking scheme is configurable, it must also be presented. Beside the current item, also an index list should be (partly) presented. The items' numbers can also be used as links to the corresponding item. Also a switch to a full list view and switches for jumping to the top and to the bottom can be added.
Attributes	• Atoms for list elements (Link) • Atoms for forward/backward function (Triggering Element) • Ranking scheme • Atoms to index view and top/bottom view (Link)
Graphical Example: Thematic help function in MATLAB 6.5[1]; the ordering scheme can be found in the column on the left, the navigation mechanism in the frame on the right	

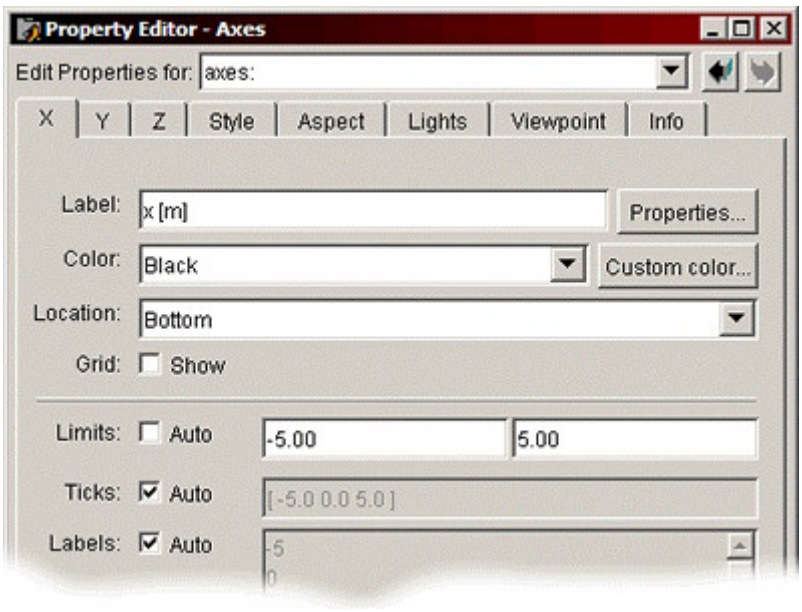
[1] The MathWorks, Inc.

Continuous Filter	
Problem	**The user needs to find an item in an ordered set**
Conditions	• The user is searching for an item in a large ordered set and may not be familiar with the exact item • The user might not be sure if the item even exists • The search term may lead to a multiple result
Solutions	**Provide a filter component with which the user can in real time filter only the items in the data that are of his interest** The filtered set is presented concurrently with the search term from the moment the user starts entering the search term. If relevant, the closest matches are high-lighted while some previous and successive items might be shown as well.
Attributes	• Matching search terms • Highlight on best match • Feedback of input data
Graphical Example: Index help function in MATLAB 6.5[1]	

[1] The MathWorks, Inc.

Preview	
Problem	**The user searches one item in a set of items and tries to find it by browsing the set. Often, other search criterions than the items name are more effective.**
Conditions	• The resources needed to present the real item in full quality may be unavailable • The user might not be able to identify the item by its name
Solutions	**Allow the user to preview the item** To save loading time, the preview can use fewer resources (resolution, sound quality…), sufficiently good for identifying the item. If the number of items is small, a preview for all items can be presented. Otherwise, the preview should only be presented for the currently selected item in combination with the name of the item. The preview should change immediately when the selection changes. The used resources also depend on the type of the previewed item.
Attributes	• Representation quality of preview
Graphical Example: Graphical representation of a folder with MS Explorer[1] in Preview Mode	

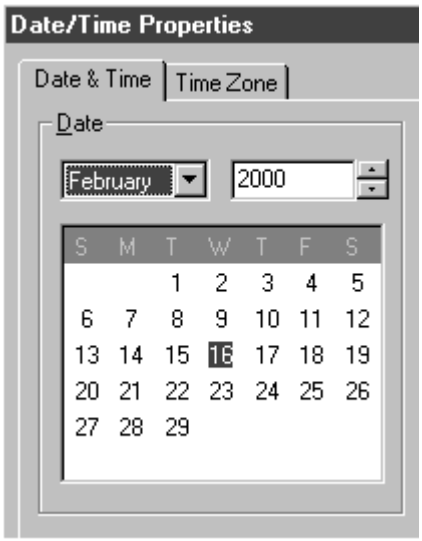
[1] Microsoft ® Windows, The Microsoft Corporation

Navigating between categories	
Problem	**The user needs to access an amount of information which cannot be put on the available representation area.**
Conditions	• The available representation area is limited • Data is often related and can be divided into categories that match the user's conceptual model of the data.
Solutions	**Group the information in several categories and allow the user to navigate between them.** Information elements are grouped in separate categories, where only one category can be selected at a time. Each category should be labelled with its name. Switching between categories should be performed with one single action from a navigation mechanism which includes all categories. This navigation mechanism should be represented separated from the categories, at the border of the representation area. At any time, the selected category and the navigation mechanism are present. Each category can host an independent interaction pattern or atoms.
Attributes	• Navigation mechanism • Different categories • Labelling of categories • Contents of categories
Graphical Example: Property Editor from MATLAB 6.5[1]; the content is thematically grouped in several tabs.	

[1] The MathWorks Inc.

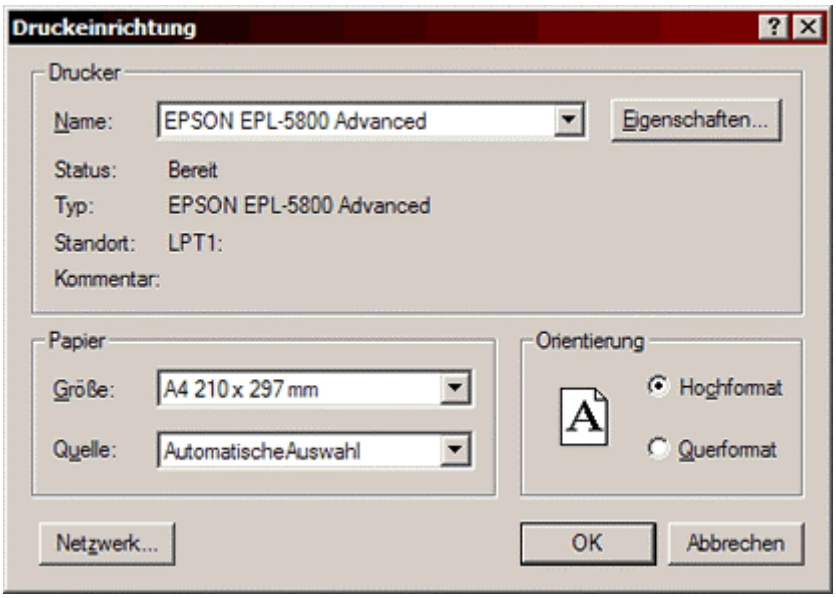
Container Navigation	
Problem	**The user needs to find an item in a collection of containers**
Conditions	• User needs to see all available containers, the content of one container and an item at the same time • Not all items need to be visible at the same time • The user may need to switch between containers
Solutions	**Split up the representation area in three regions for the containers and the final selection** Split the representation area into three regions, one for a collection of all containers, one for detailed representation of the contents of one selected container, and one for a selected item within the selected container. The selections may be used as parameters to invoked functions. Each region should be specialised to the type of content it presents. E.g. if the containers form a hierarchy, a tree structure providing selection of leaf nodes could be used. The regions should be individually browsable and resizable.
Attributes	• Regions • Resizing mechanism • Atoms (Tree Structure) for all containers and the contents of the selected container
Graphical Example: In Outlook Express[1], one container gives an overview of available mail accounts and newsgroups in three regions.	

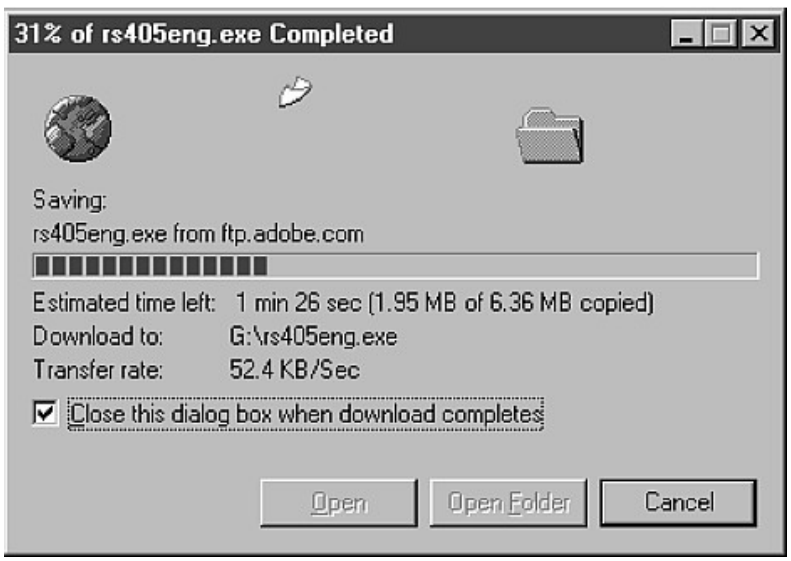
[1] © The Microsoft Corporation

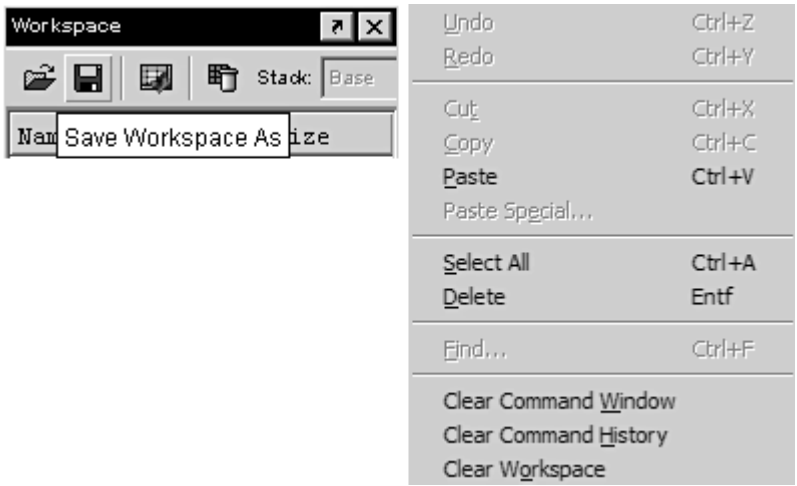
Unambiguous Format	
Problem	**The user needs to supply the application with data but may not know which type of data is required or what syntax to use.**
Conditions	• The application needs a certain syntax which might be unknown by the user. • The user strives for entry speed but also wants data to be entered correctly • Different users may expect a different syntax for the same data.
Solutions	**Only allow the user to enter data in the correct syntax** For each data element to enter, a separate area of the representational area must be provided with appropriate labelling. The data field does not allow incorrect data to be entered. The syntax can be explained with an example or a description of the input format.
Attributes	• Atom for labelling of input information (Information) • Atoms for data fields (Data Entry) • Characteristic feature of data field
Graphical Example: Time and date control panel from MS Windows[1], taken from [42].	

[1] Microsoft ® Windows, The Microsoft Corporation

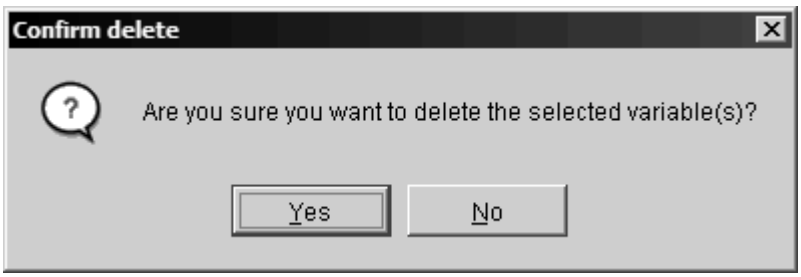
Focus/Selection	
Problem	**Users want to quickly get information about an object they see and possibly want to modify the object**
Conditions	• The user usually works on one object at a time • The user wants both an overview of the set of objects and details on attributes and available functions on the objects • The user may also want to apply a function to several objects
Solutions	**Introduce a focus in the application** The focus always belongs to one or more objects present in the interface. It must be shown by the change of one attribute of the object or by creating some additional attribute. When an object has the focus, it becomes the target for all the functionality that is relevant for the object.
Attributes	• Attribute that shows “focus on”/”focus off”
Graphical Example: From a text excerpt, some lines are selected, which is expressed by different background colour	Einer seit waz er gesiht, der ander seit waz im geschiht, der dritte von minne, der vierde von gewinne, der fünfte von grôzem guote, der sehste von hôhem muote. hie wil ich sagen waz mir geschach daz ich mit mînen ougen sach. Ich sach, daz ist sicherlîch wâr, eines gebûren sun der truoc ein hâr daz was reide unde val. ob der ahsel hin ze tal mit lenge ez volleclîchen gie. (Extract from „Meier Helmbrecht“ from Wernher der Gartenaere, Middle high german literature from the 13th century)

Frame/Grouping Layout	
Problem	**The user needs to quickly understand information and take action depending on that information. Several different objects have to share a limited amount of representational space.**
Conditions	• Objects should be positioned in a clear and structured way • Related objects should be grouped conceptually • The necessary time for perceiving the present objects should be minimised
Solutions	**Arrange objects due to a fitting pattern** Objects should not be placed randomly across the available space, but due to a fitting pattern. This pattern should be consistent or at least similar for the whole interface. Similar or related objects can be grouped. The structure of the pattern should be easy to perceive in order to give extra information about the possible places for objects (cells). Each cell can contain an interaction pattern or a atom.
Attributes	• Structure of frame layout • Cell contents according to their specific attributes
Graphical Example: Printer properties, placed in a Framework of three major cells	

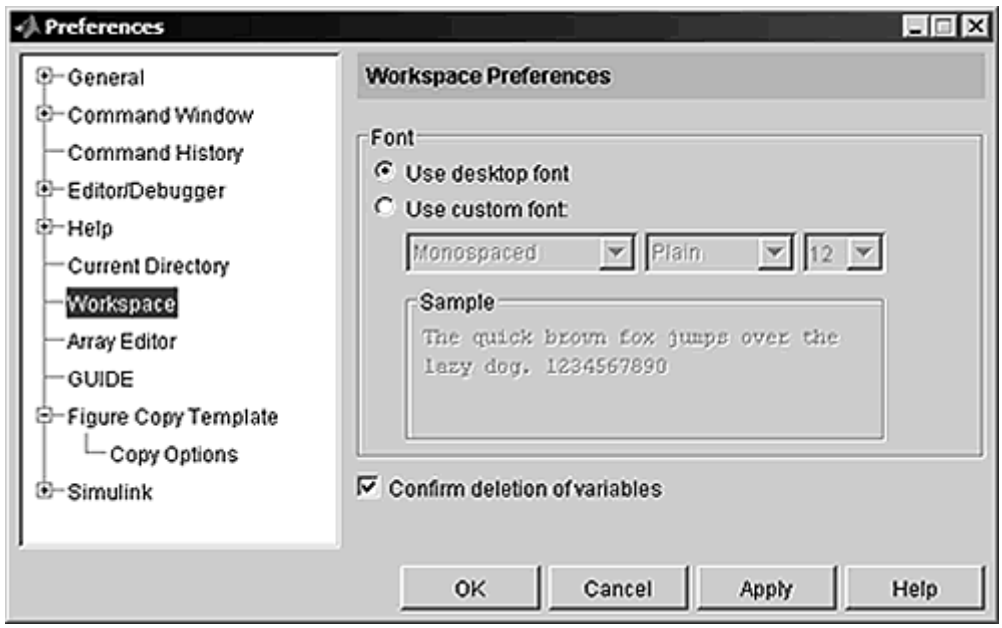
Progress	
Problem	**The user wants to know whether or not the operation is still being performed as well as how much longer the user will need to wait.**
Conditions	• The user wants to know, if the task is still being executed, what part (percentage) of the task is already executed/still left and how long the execution will approximately take. • The user wants a notification when the execution has finished • The user might want to interrupt/cancel the execution.
Solutions	**Show that the application is still working and give an indication of the progress** The feedback of progress has to be presented at a rate that shows that the application is still being performed, and should also include information about the percentage of the already executed part. The values of 0% and 100% must be mapped to a quantity which is perceivable (nearly) as absolute value, so that from any value between the minimum and the maximum of the representing quantity, the underlying percentage can be approximated by the user without effort. Additionally, the progress information can include two atoms: detailed information about the already executed part of the application (executed %, already received data, remaining data, remaining time…) and a mechanism to stop the execution.
Attributes	• Minimal and maximal value of representing quantity + quasi-continuous values in between • Atoms for stop mechanism and additional information (Triggering Elements, Information)
Graphical Example: Progress Window, Example taken from [42].	

Hinting	
Problem	**The user needs to know how to select functions, especially if they are accessible in more than one way (menu, keyboard shortcut, toolbar…)**
Conditions	• The available representation area is limited, there is no space for extra hints • The user needs a possibility to learn and discover the alternatives • The user may or may not already know the other ways to access the function • The number of ways to activate the function determines the number of possible hints.
Solutions	**Give the user hints for other ways to access the same function** When the user accesses a function in one way, provide hints for other ways to access the same function. A possibility is to use multiple labels: one label for each way the functions can be accessed. Another possibility is to give a delayed hint (represent the function's name) when user puts focus on one function for a certain amount of time (e.g. 2 seconds).
Attributes	• Labelling of functions • For delayed hint: delay time and representation of hint
Graphical Examples: Hinting within toolbar and menu structure of MATLAB 6.5[1]	

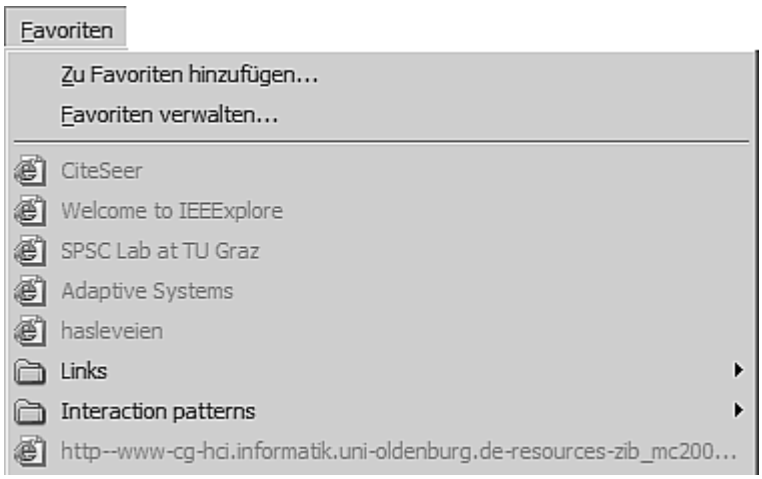
[1] The MathWorks Inc.

Warning/Message	
Problem	**The user may unintentionally cause a problem situation which needs to be resolved.**
Conditions	• Work may be lost if the action is fully completed. • The system can or should not automatically resolve the situation. • The number of ways in which the problem can be resolved must be clear. • Users may not understand the consequences of the options • The severity of the problem if it occurs must be expressed.
Solutions	**Warn the user before continuing the task and give the chance to abort the tasks.** The warning should contain the following elements: • A summary of the problem • The condition that has triggered the warning • A question asking the users whether to continue the action or take on other actions • Two main choices for the user, an affirmative choice and a choice to abort the action The warning might also include a more detailed description of the situation to help the user make the appropriate decision. The choices should be labelled with regard to their related action. Do not use Yes/No choices. In some cases there may me more than two choices. Increasing the number of choices may be acceptable, but strive to minimise the number of choices.
Attributes	• Atoms for problem summary, condition and question (Information) • Atoms for the choices (Triggering Elements)
Graphical Examples	

Shield	
Problem	**The user may accidentally select a function that has irreversible effects.**
Conditions	• The user needs protection from irreversible effects or undesired situations. • The normal operation should be unchanged.
Solutions	**Protect the user by inserting a shield** An extra protection layer is added to certain functions to protect the user from making mistakes. This protection layer consists of the question, whether a certain action should be performed, and the possibility to confirm or decline this action.
Attributes	• Atom for the question (Information) • Atoms for accept/decline (Triggering Element)
Graphical Example: The system does not allow to create two folders with the same name within the same directory.	

Preferences	
Problem	**Each user is different and prefers to do things slightly different.**
Conditions	• User may not be familiar with the possible options for adaptation • The user needs to know the consequences of the adaptations • Users will only want to change a subset of possible changes
Solutions	**Allow the user to adjust preferences** Provide choices for users which will become default settings for this user on further use. The options can often be grouped. The representation of the groups depends on their number. If the number is small, the “navigation between categories” pattern can be used. If the number of groups is high, a list browser can be used. For the collection of the options for one group, atoms and frame-layout can be used. Each option must be explained so that the users know the consequences.
Attributes	• Atoms for the representation of options
Graphical Example: Preferences Dialogue from MATLAB 6.5[1]	

[1] The MathWorks Inc.

Favourites	
Problem	**The user needs to find a regularly used item within a large set of items**
Conditions	• Regularly used items need to be at hand. • The number of contained items within a set determines the required search time. • The user is interested in the item but may use it only for a short period of time.
Solutions	**Allow the usage of favourites that point to the items of choice** A favourite is a label that points directly to the particular item. Favourites should be accessible in a minimum number of user actions, typically directly from a contextual menu. If necessary, favourites should be hierarchically ordered.
Attributes	• Label of favourite • Location of favourites
Graphical Example: Favourites from MS Internet Explorer[1]	

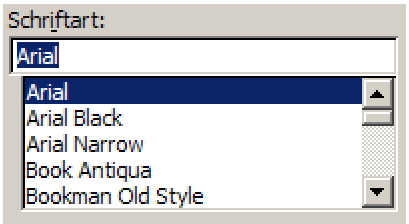
[1] © The Microsoft Corp.

Appendix II: Atoms

This section offers a set of 10 atoms for complementary usage with the generalised interaction patterns. A description of the correct usage of atoms can be found in chapter 2 of this thesis.

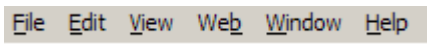
Selection	
Problem	**The user wants to select one or more items from a small number of options. One or more possibilities can be pre-selected by the application itself**.
Conditions	• The user wants information about all existing possibilities • Status information about the currently selected option must be provided
Solutions	**Grouping of selectable possibilities:** The user selects from a limited set with predefined options. The selection can be either 1 of N or M of N elements. The elements must be arranged in a way that they can be perceived as "belonging to the same set" and can be easily distinguished from all other objects. Elements are selected or deselected with a single action. If an element is selected by the user, feedback has to be given. This feedback should enable the user to detect selected elements quickly.
Attributes	• Property "Selection" • Representation of single elements • Status information (selected/deselected) = feedback information • Contextual parameter
Graphical Examples: Selectable Options within MATLAB 6.5[1]	

[1] The MathWorks Inc.

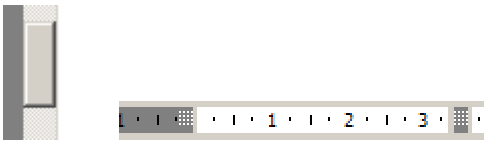
Selection with browsing	
Problem	**The user needs to select from a large number of possibilities. One of the possibilities is pre-selected by the application itself.**
Conditions	• The user wants information about all existing possibilities • Status information about the currently selected options must be provided. • To reduce complexity, the selected option is presented with the additional information: “further possibilities available”
Solutions	**Grouping of selectable possibilities, browsing tool:** The user selects from a limited set with predefined possibilities. The selection can be 1 of N elements. To view all possible elements, a browsing tool must be provided, which includes all possible elements. Within the browsing tool, serial (one element at a time) or parallel presentation (a few elements at a time) is possible. A browsing mechanism (“forward”, “backward”) must be provided. The selection of an element can be performed with a single action. After the selection, as a feedback, the selected element is presented.
Attributes	• Property “Selection with browsing” • Status information (presentation of the currently selected element) • Representation of the elements within the browsing tool • Browsing mechanism (forward- and backward functions, number of presented elements/time) • Contextual parameter
Graphical Example: Selectable Fonts within MS Word[1]	

[1] Microsoft ® Word, The Microsoft Corp.

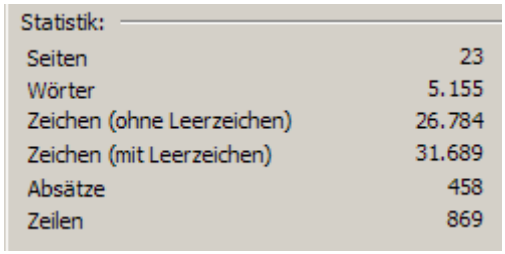
Triggering Elements	
Problem	**The user wants to start a certain function within the application (e.g. to have settings stored or functions started) .**
Conditions	• The user wants information about the effect of the triggering element.
Solutions	**Usage of a triggering element** A triggering element is linked to a certain function which is executed once the element is activated by the user. The related function and the domain where this function will be applied has to be labelled clearly or should be obvious within the context. If the triggering element is activated, feedback has to be given.
Attributes to map	• Property “Triggering Element” • Information about related function • Feedback information • Contextual parameter
Graphical Examples	

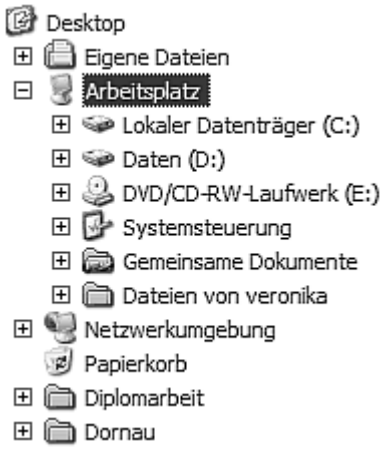
Link	
Problem	**The user wants fast access to a specific point of the application. He has knowledge about different functional parts of the application and wants to be able to reach this parts quickly.**
Conditions	• The user wants information about the location
Solutions	**Defining an active area as link** A link gives the possibility to jump immediately to a different function within the application. The location where the link leads to has to be labelled short and unambiguous.
Attributes to map	• Property “Link” • Labelling of the link destination • Contextual parameter
Graphical Examples	

Data Field	
Problem	**The user wants to enter data to the application via keyboard. At this level, the entered data is unspecified and not restricted to a certain format (text, numbers).**
Conditions	• The user wants to be certain, that any input is stored • The user must be able to correct mistakes
Solutions	**Defining an active area as data field** For meaningful processing, the user can enter data only at defined spots of the user interface. A specific area is defined as data field. After activating the data field by a single action, the user can enter any type of data with the keyboard. Feedback of the input data has to be given immediately. Additionally, feedback is needed to inform the user whether the data field is activated or not.
Attributes to map	• Property “data field” • Feedback of input information • Feedback active/not active • Contextual parameter
Graphical Example	

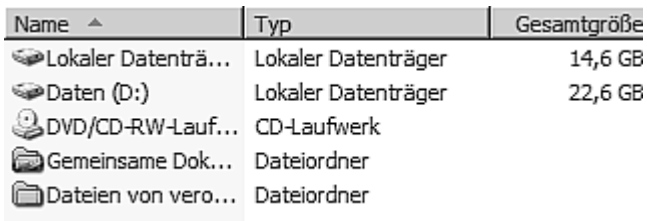
Continuous increase/decrease of a value	
Problem	**The user wants to change the size of a value according to some estimation of a desired value continuously.**
Conditions	• The size of the value must be mapped to some movement performed by the user • The direction of the mapping must be intuitive for the user
Solutions	**Continuous increase/decrease of a value** The user can increase or decrease the current value continuously by selecting a marker which represents the current value, moving it into the desired direction and then releasing it. The movement can be performed with the aid of "quasi-analogue" input devices (e.g. mouse, joystick…). The direction of the mapping movement – change of the value must be chosen depending on the context, and feedback has to be given immediately.
Attributes to map	• Property "continuos increase/decrease" • Feedback of change in value
Graphical Examples	

Discrete increase/decrease of a value	
Problem	**The user wants to change the size of a value discretely by performing a repeated interaction (mouse, keystroke).**
Conditions	• The size of the value must be mapped to a certain input device • For every possible direction of change, a different input process is needed
Solutions	**Discrete increase/decrease of a value** The user can increase or decrease the current value in discrete steps by repeatedly hitting a button on keyboard or mouse. The used stepsize must be chosen due to practical viewpoints, and feedback must be provided.
Attributes to map	• Property "discrete increase/decrease" • Feedback of change in value • Contextual parameter
Graphical Examples	

Information	
Problem	**The user may need additional information about the status of the application or the result of a query.**
Conditions	• The user needs detailed information about the status of the application. • The information and the format of this information may vary among a wide range of possibilities • Possible feedback format of user entry
Solutions	**Provide information** If exact feedback has to be given and the feedback information and its format is allowed to vary among a wide range of options, the only way to present this information is to directly present the contents in an appropriate way (labelling, spoken hint…).
Attributes to map	• Property “information” • Information representation • Contextual parameter
Graphic Example	

Tree Structure	
Problem	**The user wants an overview of organised data filing**
Conditions	• An organisation scheme for files and folders must be provided • The user wants information about the location of certain data and containing folders. • Often, functions of an application are organised in a way similar to data file organisation
Solutions	**Organise data in tree structures** The tree-like organisation scheme of data is well known and often used. The location of the data file and the containing folder can be extracted from the root of the tree and the passed knots. A distinction between knots that can contain further knots (= parent folders) and knots that cannot contain further knots (= files or folders) must be made. The user must be able to query the content of one knot with a simple action. Also status information whether the knot contains data or not can be provided.
Attributes to map	• Property “tree structure” • Presentation of different types of knots (files, folders) • Presentation of status (whether folder contains files or not) • Query mechanism • Contextual parameter
Graphical Example Folder hierarchy of MS Explorer[1]	

[1] Microsoft ® Windows, The Microsoft Corp.

List	
Problem	**The user wants an ordered overview of larger amounts of data**
Conditions	• An organisation scheme for data elements must be provided • The user may want to customise the sorting scheme
Solutions	**Organise data in lists** A list of data which can be sorted according to different properties of the data elements is a very common used organisation scheme (e.g tables). The properties of each data element must be visible, and the user must be able to switch between the sorting possibilites.
Attributes to map	• Property “list” • Presentation of list elements • Presentation of properties • Sorting scheme • Contextual parameter
Graphical Example A list presents the content of a folder in MS Explorer[1]	

[1] Microsoft ® Windows, The Microsoft Corp.

Appendix III: Test Details

Appendix III gives further information about the two listening test described in chapter 4 of the thesis. For both tests, the tasks the participants had to perform and selected results of the test evaluation are presented.

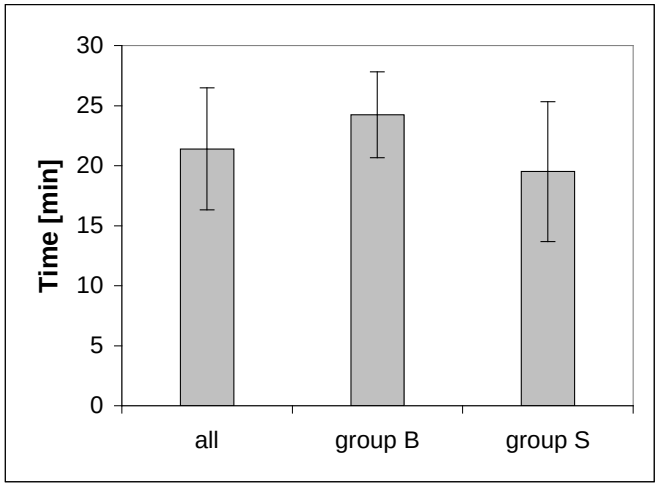
III/1. Results of test 1

Task	Description	Results
1	Opening the dialog "file – new" and entering data in four text fields. The functionality of each textfield had to be found via F1-function (name/street/village/date)	4 correct textfields
2	In the "handling"-submenu, the terms of delivery (dialog "delivery") and payment (dialog "payment") had to be chosen among a list of two possibilities for each. The meaning of the list elements head to be found via F1-function. For payment, a fictitious credit card number had to be entered.	2 correct lists correct number
3	From the "goods"-submenu, participants had to choose eight correct items out of four lists (fruit, milk products, beverages, bread).	4 correct lists
4	Participants had to enter two specific items into the search function and then add them to their shopping list	2 correct results
5	Finally, participants were asked to close the application with the "close"-menu	correct result

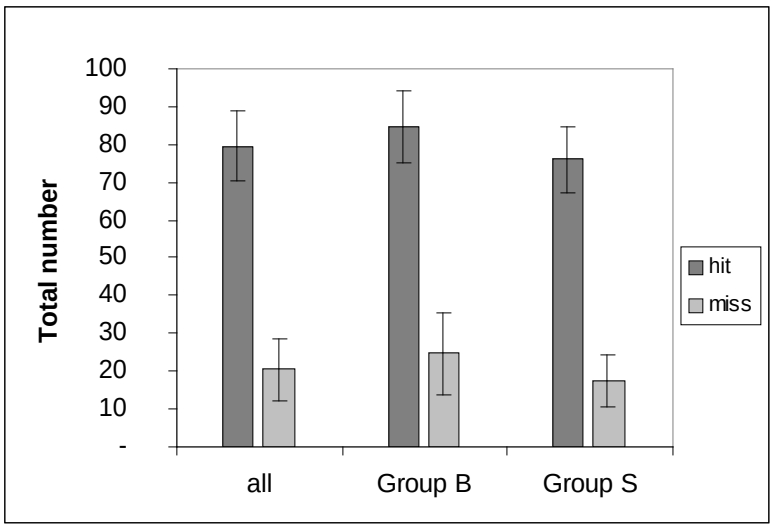
[Tab 5] Tasks to fulfil, first test

The required time for the fulfilling of all tasks was compared for all users. An average value was computed over the results of all participants. Additionally, an average was computed over the members of group B and over the members of group S. The averages are shown in [Fig 21]. Group B was a bit slower than group S, but the difference lies within the variance of data.

Comparing the total number of clicks between the participants is a measurement of the orientation of the participant within the menu structure. The more clicks a user caused, the more unnecessary actions were performed. The minimum number of required hits (clicks which met their target) to perform the complete task without any mistake was 57. An increased number of necessary hits for the participants is caused by selecting items more than just once, entering the wrong submenu, causing alerts which have to be confirmed etc. [Fig 22] shows the group averages of the total number of hits and misses. Again, the performances of both groups are comparable.

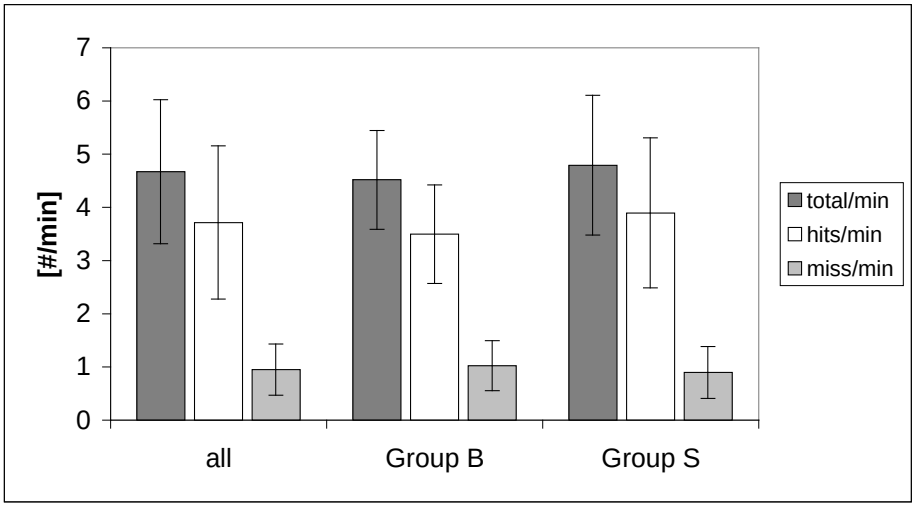


[Fig 21] Total test time, group averages



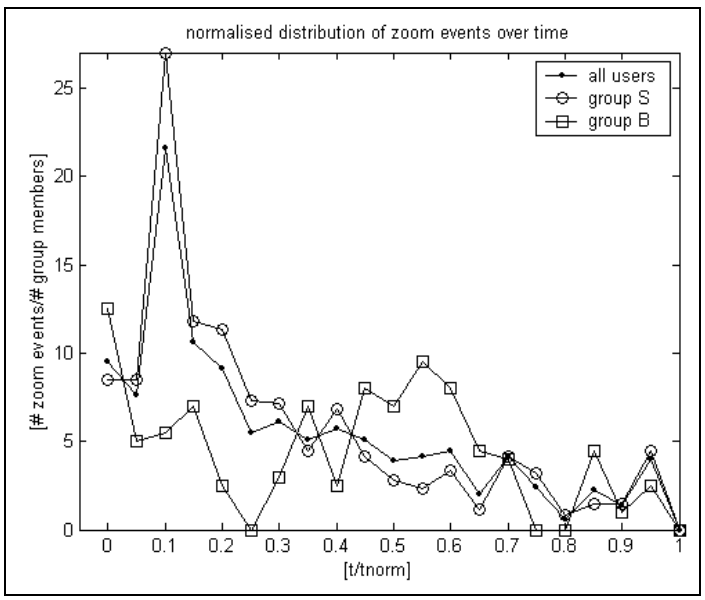
[Fig 22] Number of clicks (hits and miss), averages

As an additional measurement of performance, the number of selection events per minute is calculated by simply dividing the total number of occurrence of the different events by the total required time of each user. [Fig 23] shows the selection events/min as average for each group. It is remarkable, that the averages seem to be almost constant over the different user groups, although the variance is bigger for group S.



[Fig 23] Number of selective events per minute

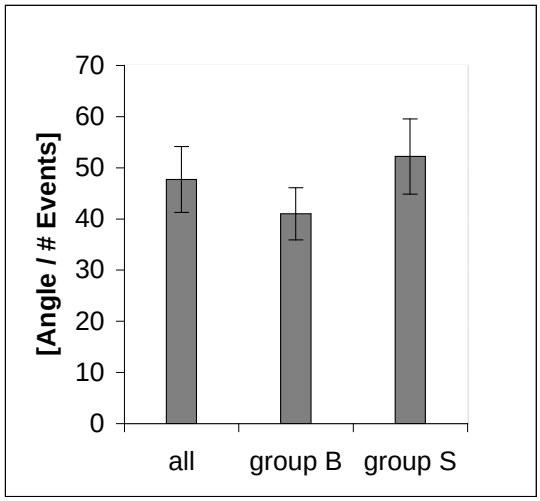
The first test has shown, that the zooming function or acoustical lens which should enable focusing on selected sources was difficult to handle. Throughout the test, some participants completely stopped zooming and replaced it by the use of the helping function. To get more precise information about the usage of the zoom function, a distribution of this function was calculated over the normalised performance time for each user. For this purpose, a histogram with a resolution of 20 divisions was used. The histograms were added up for group S, group B and for all users and normalised to the number of group members to make the amounts comparable. [Fig 24] shows the distribution of zoom events over time. The users with normal seeing ability called the zoom function more often than the blind users. Generally, the usage decreases with time.



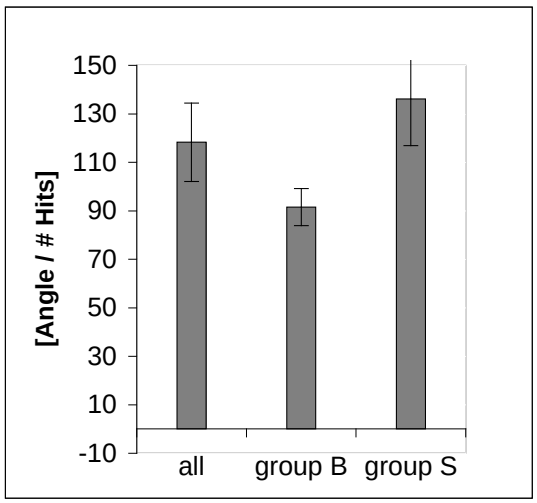
[Fig 24] Normalised distribution of zoom events over time

To get information about the head movement which was required to reach a certain source, the relation between the angle covered by head rotation and the number of events was calculated. For this purpose, the differential angle was used. Calculations were performed for one task only, namely task no. 3.

[Fig 25] shows, that the covered angle and with it the required head movement to reach a certain goal lies within the range of the presentation area of auditory icons (± 60°). This calculation is performed for all kind of events, including missed hits and help functions, not including the zoom function. [Fig 26] shows the same calculation performed for hits (selections that reached their goal) only. Comparing both calculations it is remarkable, that half the way is wasted with help events or misses.



[Fig 25] Covered Angle per Event



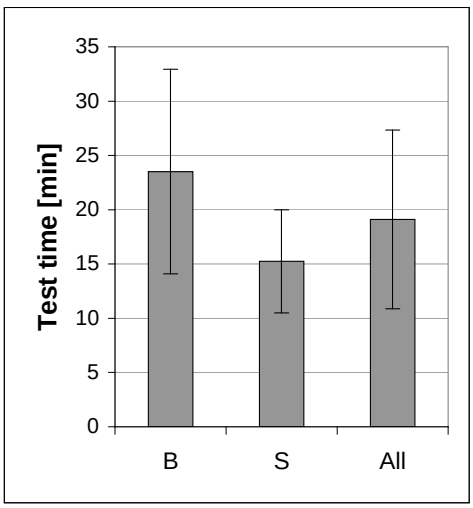
[Fig 26] Covered Angle per Hit

III/2. Results of test 2

Task	Description	Results
1	How many files does the folder music contain? (The participants were told the location of the folder music).	1 correct answer (verbal)
2	Find out the size of the file music1 with the aid of the properties-window or by changing the view-mode from “list” to “details” .	1 correct answer (verbal)
3	Find the folder windows and name its position without the help of the search function.	1 correct answer (verbal)
4	Create a new folder directly on the hard disc C:.	1 new folder on correct location
5	From the folder letters , cancel the file with the name letter4	Report of deleting process
6	Undo task no. 5 with the aid of the menu structure or with the toolbar.	Report of undo process
7	Directly on the hard disc C:, a file with the name video4.mpeg can be found. Move this file to the folder videos .	Report of move/copy process

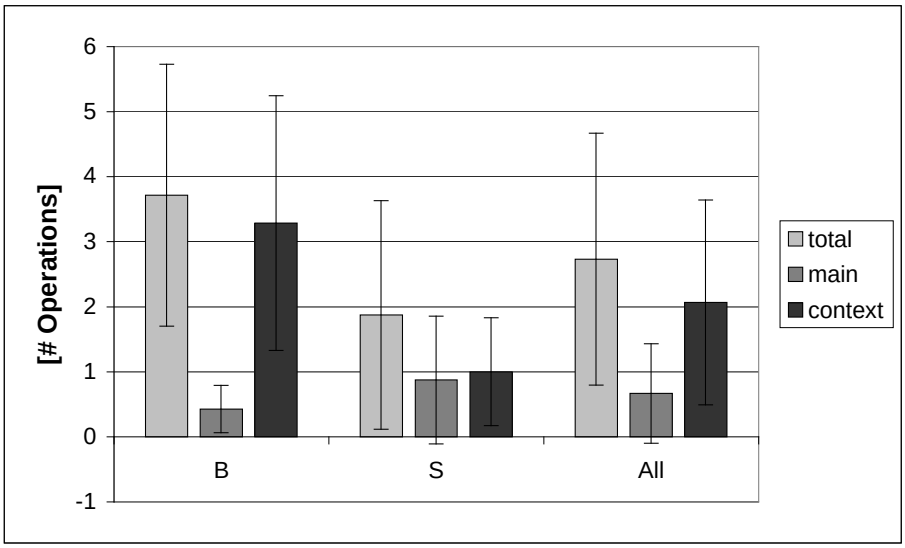
[Tab 6] Tasks to fulfil, second test

The required time for the fulfilling of all tasks was calculated as average value for the three groups ([Fig 27]). The results show, that on average, group B needed more time to fulfil the test, but also that the standard deviation is high, which was mainly caused by two of the blind participants.



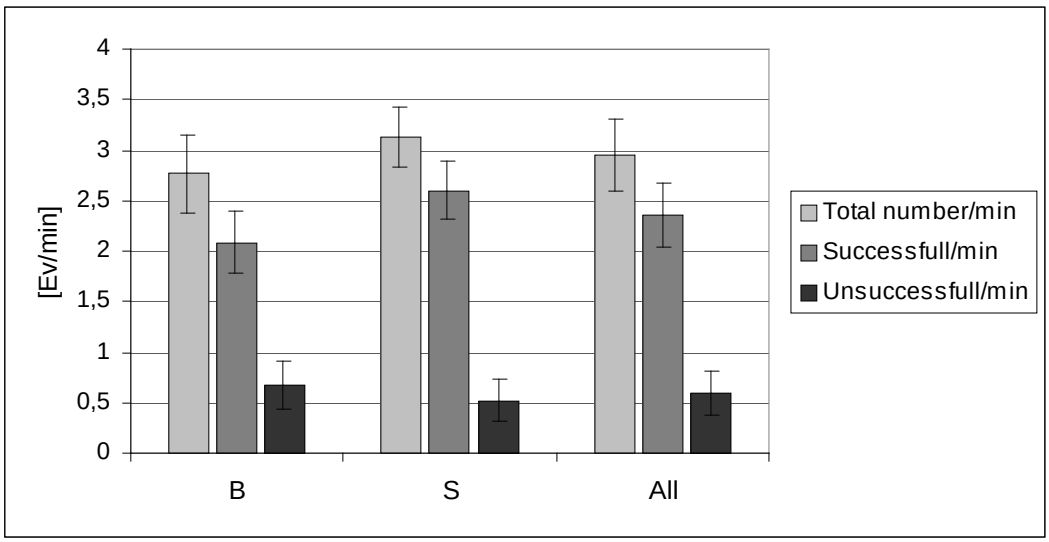
[Fig 27] Total test time, group averages

Another measurement of performance is the number of unintentionally performed menu operations. The selection of a menu header which is not accompanied by the selection of one of its submenu items is counted as unintentional selection. Note, that this value also includes searching processes of users who did not know in which main menu the desired submenu item can be found. The number of intentionally and unintentionally performed menu operations can be seen in [Fig 28]. The graph shows that for group B, the total number of unintentional menu operations is higher than for group S, but the majority of these operations is caused by unintentionally opening the context menu, which can be explained by mistaking the functionality of right-clicking and left-clicking. The number of unintentional operations within the main menu is much lower for group B than for group S.



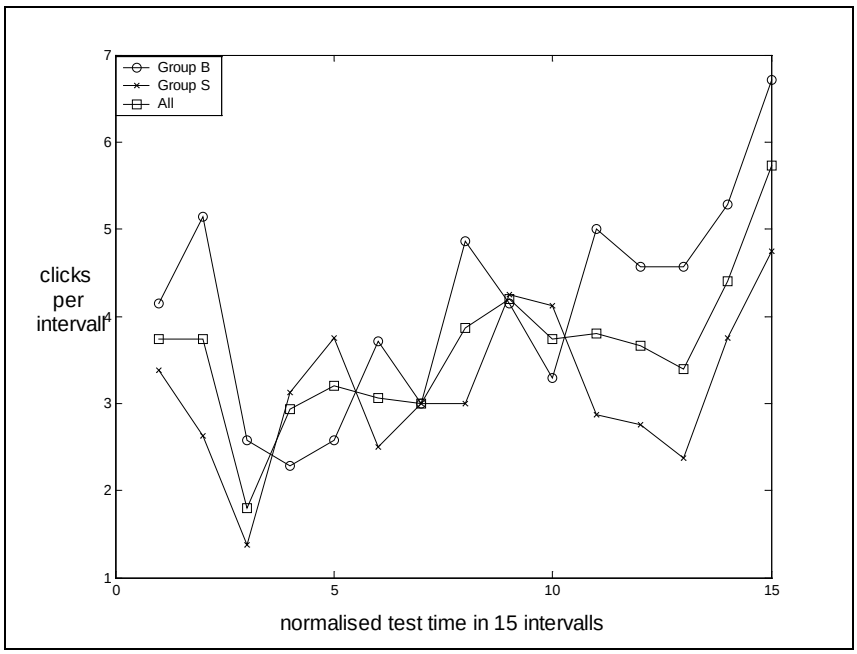
[Fig 28] Number of unintentional performed menu operations

Each user interaction caused a selective event. These events, put into relation with the total performance time of the participants, can be used to get a comparable value for the working speed [Fig 29], expressed as number of selective events per minute. Between the working speed of the groups, there is only a small difference.



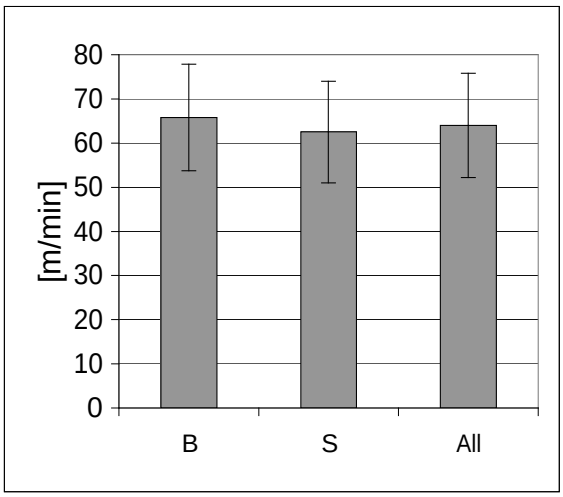
[Fig 29] Number of selective events/minute

A measurement for a possible improvement of the working speed with increasing test time is the distribution of selective events over the total test time of each user. The time index of the selective events was normalised to the total test time of each participant. Then, the time indices were added up for the groups B, S and for all users. A histogram was calculated and normalised to the number of group members ([Fig 30]). An improvement of the participants is equivalent to an increase of the number of selective events with time. Although the complexity of the tasks increased with time, an improvement can be seen for all groups.



[Fig 30] Distribution of selective events vs. time

To compare the velocity of the joystick movement between the three groups, the travelled distance in meters was divided by the total test time. The results in [m/min] can be seen in [Fig 31]. The graph shows nearly equal velocity for the three groups, although there is a high difference between the group members represented by a high standard deviation.



[Fig 31] Velocity of the joystick movement [m/min]

6. References

[1] Arons B.: “A Review of the Cocktail Party Effect”, Journal of the American Voice I/O Society, vol 12, pp 35-50, 1992

[2] Badeley A.: “Human Memory: Theory and Practice”, Lawrence Erlbaum Associates, 1990

[3] Barr P., Biddle R., Noble J.: “A Taxonomy of user-interface metaphors”, School of Mathematical and Computing Sciences, Victoria University of Wellington, 2002

[4] Bly S.: “Presenting Information in Sound”, Proceedings of the CHI Conference on Human Factors in Computer Systems, pp. 371-375, 1982

[5] Bregman A.S: “Auditory Scene Analysis – The Perceptual Organisation of Sound”, Bradford Book, 1994

[6] Brewster S.A.: “Providing a structured method for integrating non-speech-audio into human computer interfaces”, Dissertation, 1994

[7] Brock D., Ballas J.A., Stroup J.L., McClimens B.: “The design of mixed-use, virtual auditory displays: recent findings with a dual-task paradigm”, ICAD Proceedings 2004

[8] Brungart D.S, Ericson M.: “Design considerations for improving the effectiveness of multi-talker speech displays”, ICAD Proceedings 2002

[9] Buxton W.: “Introduction to this special issue on nonspeech audio”, Human Computer Interaction, 4(1), 1989

[10] Carroll J.M, Mack R.L., Kellogg W.A.: “Interface Metaphors and User Interface Design”, in *Handbook of Human-Computer Interaction*, Elsevier Science Publishers, 1988.

[11] Cherry E.C.: “Some experiments on the recognition of speech, with one and with two ears”, Journal of the Acoustical Society of America 25, pp 975-979, 1953

[12] Coomans M.K.D., Timmermans H.J.P.: “Towards a taxonomy of virtual reality user interfaces”, Proceedings of the International Conference on Information Visualisation 1997

[13] Darwin C.J.: “Auditory grouping and attention to speech”, Webpaper, Experimental Psychology, University of Sussex, Brighton

[14] Deutsch D.: “Auditory pattern recognition”. In K. R. Boff, L. Kaufman, & P. Thomas (Eds.), *Handbook of perception and human performance*, 1986

[15] Donald A. Norman: “The Design of Everyday Things”, Doubleday, 1988

[16] Edwards W.K., Mynatt E.D.: “The Mercator project – a non-visual interface to the X Window System”, in *The X Resource* #7, 1993.

[17] Frauenberger Ch., Höldrich R., da Campo A.: “A generic, semantically based design approach for spatial auditory computer displays”, ICAD Proceedings 2004

[18] Gabbard J.L., Hix D.: “A Taxonomy of usability characteristics in virtual environments”, Departement of Computer Science, Virginia Polytechnic Institute and State University, 1997

[19] Gaver W.: “Auditory Icons: Using sound in computer interfaces”, Human Computer Interaction, 2(2), pp. 167-177, 1986

[20] Gaver W.W., “The SonicFinder: An interface that uses auditory icons”, Human Computer Interaction, 4(1), pp. 67–94, 1989

[21] Gerzon M. A.: “Ambisonics in multichannel broadcasting and video”, Journal of the Audio Engineering Society, vol. 33, pp. 859-871, 1985

[22] Hunt A., Hermann T.: “The importance of interaction in sonification”, ICAD Proceedings 2004

[23] Kubovy M., van Valkenburg D.: “Auditory and visual objects”, Tutorial, Department of Psychology, University of Virginia, 2003

[24] Kubovy M.: “Concurrent-pitch segregation and the theory of indispensable attributes”, in Kubovy M. & Pomerantz J.: “Perceptual organisation”, Lawrence Erlbaum, 1981

[25] Lakoff G., and Johnson M., “Metaphors We Live By”, the University of Chicago Press, 1980.

[26] Lennox P.P., Vaughan J.M., Tony Myatt T.: “3D Audio as information-environment: Manipulating perceptual significance for differentiation and pre-selection” , ICAD Proceedings 2001

[27] Ludwig L.L., Pincever N., and Cohen M.: “Extending the notion of a window system to audio”, Computer Volume 23, Issue 8, pp 66-72, 1990

[28] McGookin D.K, Brewster S.A: “An investigation into the identification of concurrently presented Earcons”, ICAD Proceedings 2003

[29] Mountford S.J., Gaver W.W.: “Talking and Listening to Computers. The Art of Human-Computer Interface Design”, edited by Brenda Laurel. Addison-Wesley Publishing Company, 319-334, 1990

[30] Mynatt E.: “Transforming Graphical Interfaces into Auditory Interfaces”, Dissertation, 1995

[31] Mynatt E.D, Edwards W.K: “Metaphors for nonvisual computing”, taken from *Extraordinary Human-Computer Interaction*, Cambridge University Press, 1996

[32] Noisternig M., Musil T., Sontacchi A., Höldrich R.: “A 3D real time rendering engine for binaural sound reproduction”, ICAD Proceedings 2003

[33] Pitt I.J., Edwards A.D.N.: “Pointing in an auditory interface for blind users”, IEEE 1995

[34] Pittarello F., Celentano A.: Panphonen: “A spatial enhanced audio interface for blind users”, IEEE 2001

[35] Rasch R.A., “The perception of simultaneous notes such as in polyphonic music”. Acoustica 40 pp. 21-33, 1978

[36] Reuter C.: „Die auditive Diskrimination von Orchesterinstrumenten“. Lang, 1996

[37] Sawhney N., Murphy A.: “Designing Audio Environments – Not Audio Interfaces”, Graphics, Visualization, and Usability Center, School of Literature, Communication, and Culture, GIT

[38] Shinn-Cunningham B., Ihlefeld A.: “Selective and divided attention: Extracting information from simultaneous sound sources”, ICAD Proceedings 2004

[39] Shinn-Cunningham B.: “Speech intelligibility, spatial unmasking and realism in reverberant spatial auditory displays”, ICAD Proceedings 2002

[40] Smither J: “Short term memory demands in processing synthetic speech by old and young adults”. Behaviour and Information Technology, 12(6), pp. 330-335, 1993

[41] Sumikawa D. A., Blattner M., Joy K., Greenberg R.: “Guidelines for the Syntactic Design of Audio Cues in Computer Interfaces”. Technical Report No. UCRL 92925. Lawrence Livermore National Laboratory, 1986.

[42] van Welie M., Traetteberg H.: „Interaction Patterns in User Interfaces“, Webpaper, 2000

[43] Walker A., Brewster S.A., McGookin D., Ng A.: “Diary in the sky - A spatial audio display for a mobile calendar”, The University of Glasgow, 2001

[44] Walker B.N., Lane D.M: “Psychophysical scaling of sonification mappings: A comparision of visually impaired and sighted listeners”, ICAD Proceedings 2001

[45] Wenzel E., Arruda M., Kistler D.J., Wightman F.L.: „Localization using nonindividualized head-related transfer functions“, in Journal of the Acoustical Society vol 94, pp. 111-123, 1993

[46] Williams S.: “Perceptual principles in sound grouping”, ICAD Proceedings 1992

This document was created with Win2PDF available at http://www.daneprairie.com.
The unregistered version of Win2PDF is for evaluation or non-commercial use only.