Chapter 3

Interaction Framework

The goal is to provide an Interaction Framework to facilitate and guide the design of Collaborative Distributed Virtual Environment applications. This thesis focuses on projection based Virtual Environments (VE), where the collaboration can reach immersive face-to-face communication with the computer being the transparent medium. In this chapter a Human-Computer-Human (H-C-H) model and an H-C-H interaction framework is presented. This model is first used to design a CVE in chapter 4 and is evaluated in chapter 6.

3.1 The Human-Computer-Human Model

Currently the various modalities of interaction between humans and computers are under investigation in the *HCI* and *CHI* research community. Unfortunately in most applications the Computer-Human interaction is discussed separately from the Human-Computer interaction. Applications where these topics are addressed together are mostly groupware applications [8, 32]. In order to discuss *HC* and *CH* interaction together a Human-Computer-Human Model is proposed. Within this model the following interactions are observed:

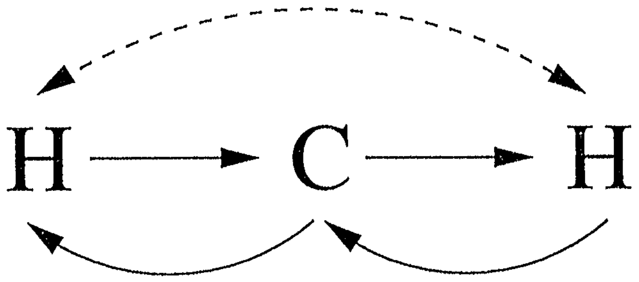


Figure 3.1: Interaction within the Human-Computer-Human Model. This model unifies the various modalities of interaction between humans and computers.

- $H \rightarrow C$ *Flow* : User input
- $C \rightarrow H$ *Flow* : Computer output
- $H \rightarrow H$ *via C Flow* : Inputs by one user are going to be displayed by the computer as outputs to the other user.

It is obvious that *direct* human to human interaction cannot happen in distributed environments. Especially when interacting with another user within a Virtual Environment the computer has to mediate and regulate this interaction. The computer as a mediator needs to set up a communication link like a video/audio connection and needs to manage the user's input and provide it as output to the other user. This follows from the fact that the interaction between human and computer is of a bidirectional nature. The computer waits and reacts on the users input queue, processes this input and then displays the reaction appropriately.
Now when looking at Figure 3.1 again it becomes clear that the Human-Human interaction can be described completely by a reduced bidirectional $H \leftrightarrow C$ chain. Although this chain is a subset of the *H-C-H* model it is possible to see that the computer processes similar inputs at both sites and presents the corresponding outputs to both sites appropriately. In other words the computer has to become omnipresent but transparent to the users.
The users of Virtual Environments do not need to know what computer hardware, what software or what communication mechanisms generate the VE. The interfaces at both sites need to be capable of hiding these from the users. As the computer becomes invisible the users have to have the feeling of communicating and interacting with another person as if face-to-face assuming an appropriate representation form for the remote user. Therefore the goal is to design distributed Collaborative Virtual Environments with interfaces in a way that the $H \leftrightarrow H$ interaction can happen as effective as possible.
As the computer is the mediator and regulator for this Human-to-Human interaction, *C* has to be optimized. The way to do this is optimizing the user interfaces as the interfaces for the $H \rightarrow C$ inputs and optimizing the viewer as the interface for the $C \rightarrow H$ outputs. In order to optimize these interaction flows within the *H-C-H* model, influence factors should be determined. These influence factors are responsible for making a user feel immersed within a Collaborative Virtual Environment and being able to communicate and interact with another person as if face-to-face. To consider these influence factors that come in play when creating distributed, Collaborative Virtual Environments, a taxonomy[1] of these influence factors is developed [47, 49, 50].

[1]The term "Taxonomy" is derived from the Greek word taxon. It is used in the science of biological classification. The taxonomist creates from a varied array of organisms a hierarchy

This developed taxonomy is not a hierarchy of classified VEs or CVEs but

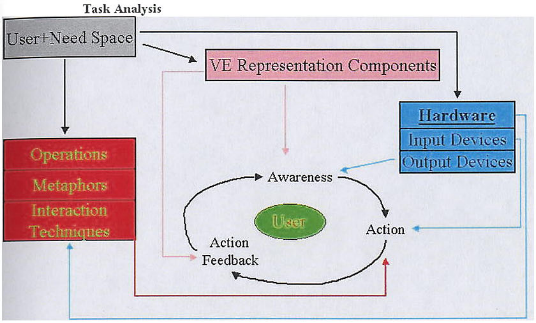


Figure 3.2: The VE and CVE design model.

a hierarchy of classified influence factors that show an impact on the design process of VEs and CVEs (see Figure 3.2). With the help of this taxonomy of influence factors it is then possible to develop a VE/CVE design model which supports the VE/CVE designer to consider the large amount of these influence factors. Thereby the design model also shows the dependency of the influence factors and enables to simulate the appearance of the VE/CVE early in the design process. The input for this simulation delivers the requirement engineering process that uses the task description and task analysis which determine the User+Need Space. An example for such a simulation is given in section 3.8.
The reason for the development of a new VE/CVE design model instead of taking an existing one is that the existing models developed in the CSCW and HCI community do not pay enough attention at Human-to-Human communication and collaboration in large scale projection based Virtual Environments. Bowman et al., for example, developed a taxonomy of different techniques

of groupings that have an orderly relationship to each other. The term is also used by Hix et al. [44] and Bowman et al. [11] for the categorization of usability characteristics and interaction techniques.

concerning the three tasks navigation/locomotion, selection and manipulation. This taxonomy enables the VE designer to find the interaction technique best-suited for a given task. The orderly relationship between the classified techniques is obtained according to usability issues taking into account user input devices, tasks and others.
The taxonomy has been developed for and evaluated in HMD based systems (Head Mounted Displays) but not in projection based displays. Whether this taxonomy is still valid for projection based display systems has to be proven in future.
As already mentioned, the existing taxonomies and VE design models do not consider collaboration and human-to-human communication aspects sufficiently which makes them useless for this work. Hence, it justifies the development of a new taxonomy of influence factors and with this the development of a new VE/CVE design model.
Therefore the utility of the taxonomy is considered rather than its absoluteness or completeness. The objective is to facilitate guided design of applications for supporting team work in CVEs [47].
One way to verify the generality of the approach is through the process of categorization, which allows to understand the low-level makeup of interaction techniques. This categorization may also lead to new design ideas. For designing the correct VE and choosing the most appropriate interaction technique using the H-C-H model, first the user tasks are specified and analysed as described in the following section. This task analysis determines the User+Need Space.
Also software engineering approaches typically use task analysis methods [30], which are concerned with the entities and the processes or tasks that need to be implemented in software. In a similar manner the User Task Analysis in this approach is concerned with the virtual data and representation components which can be considered analogous to the entities. Since this User Task Analysis is primarily focussing on collaboration and human-to-human communication, the specification of the VE/CVE is composed of the awareness-action-feedback loops and the metaphors, operations and interaction techniques which are implementing them.

3.2 User's Task Description and Analysis

Figure 3.3 shows that the approach starts with a *User's Task description (UTD)*. For example a task description could be:

Assume two users who want to connect two wooden laths. They use two ham-

mers and a box of nails. For pulling nails that are wrong pound into the wood they use a pair of pliers. They stand on top of a roof. Both laths have firstly to be connected so that they tower above the roof. Then they have to be attached to the roof. The laths are very heavy and can only be handled using both hands. The interaction space on the roof is reduced to two square meters. One user holds the wooden laths and the other user pounds the nails with the hammer or uses the pair of pliers. The users carry their tools (hammers and the box of nails) because there is not enough place to deposit them on the roof.

This description provides information about the number of users involved in

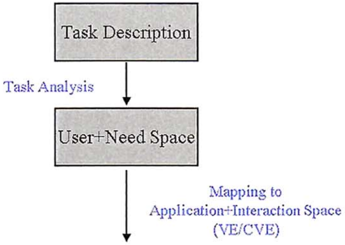


Figure 3.3: First level of the VE and CVE taxonomy graph.

the task, the type of material and the tools they use. It describes where the users stand and how they work together. This information is extracted following the *User's Task Analysis (UTA)*. This UTA determines the *User+Need Space (UNS)* which itself is the originator of the flow within the taxonomy graph. The UNS contains and groups the information extracted by the User Task Analysis of the User Task Description (UTD). It is recommendable to do an extensive description and analysis of the user's task in order to find out how the user's need can be satisfied [47, 48]. From this thesis' point of view most of the Virtual Environments lack the addressing of user needs and thus result in a poor user satisfaction and usability [14, 50, 54].
Now it is possible to do an UTA of the UTD. The information extracted by the UTA facilitates the procedure of defining a User+Need Space. For the example described above the extracted information concerns:

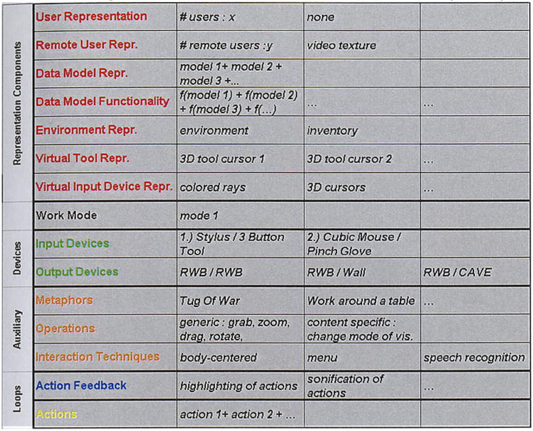
- **Participants**

- **Objects**
- **Tools**
- **Objectives**
- **Constraints**
- **Actions**
- **Reasons for Cancellation**
- **Results**

With the help of this extracted information a User+Need Space (UNS) and its content is defined as described in the following sections.

3.3 Mapping from User+Need Space to VE and CVE

The User+Need Space is represented by an array-like representation format (see Figure 3.4).
This representation is more suitable for the purpose of this thesis since it provides an easier mapping between the requirements of the UNS and the features of the Virtual Environment under design. The first seven features denote representation components. In addition to the number of local and remote users their corresponding representations are included.
Although the UNS in Figure 3.4 is a UNS template, examples of different possibilities of realizations are added for clarification. When working two-handed, different input device combinations are shown, such as a combination of a stylus and a three button tool or the combination of a pinch glove and a cubic mouse (see section 3.4). These and other combinations are not obligatory, they are only illustrating the usage of the UNS array. Also the items belonging to the operations, metaphors and interaction techniques in the auxiliary section of the array are only of illustrating nature and show that more than one item could be taken under consideration. Thereby, if in the rows appears an enumeration, the first item or combination is interpreted as the most appropriate one. Then the application designer would choose one of the suggestions. If there is no enumeration the row represents a list of items that belong together. Then all have to be taken under consideration within the application design. The next sections describe the items of the UNS in more detail.



Representation Components	User Representation	*# users : x*	*none*	
	Remote User Repr.	*# remote users :y*	*video texture*	
	Data Model Repr.	*model 1+ model 2 + model 3 +...*		
	Data Model Functionality	*f(model 1) + f(model 2) + f(model 3) + f(...)*	...	...
	Environment Repr.	*environment*	*inventory*	
	Virtual Tool Repr.	*3D tool cursor 1*	*3D tool cursor 2*	...
	Virtual Input Device Repr.	*colored rays*	*3D cursors*	...
	Work Mode	*mode 1*		
Devices	Input Devices	*1.) Stylus / 3 Button Tool*	*2.) Cubic Mouse / Pinch Glove*	
	Output Devices	*RWB / RWB*	*RWB / Wall*	*RWB / CAVE*
Auxiliary	Metaphors	*Tug Of War*	*Work around a table*	...
	Operations	*generic : grab, zoom, drag, rotate,*	*content specific : change mode of vis.*	
	Interaction Techniques	*body-centered*	*menu*	*speech recognition*
Loops	Action Feedback	*highlighting of actions*	*sonification of actions*	...
	Actions	*action 1+ action 2 + ...*		

Figure 3.4: The User+Need Space array-like representation.

3.4 Input/Output Device Combination and Work Mode

In this section the focus is on the design guidelines for combining input and output devices. It is obvious that not all six DOF input devices for interaction and output devices can be combined. For example, it is hard to use a Cubic Mouse together with a stylus in a CAVE-like display system if the stylus needs to be used frequently. The reason is simply that for using the Cubic Mouse the user needs both hands which results in putting other input devices away. Combining these input devices with for example the RWB as output device the user has the possibility of putting unused devices back on the table of the RWB. But of course this is not the only reason for choosing a certain type of input/output combination. The selection of the devices is mainly influenced by other factors. Most important factor for the selection of an adequate output device is the amount of users who work together at the same site as well as the size of the data model. The most adequate display system for an architect who shows the pre-visualized interior of a building to the client might be a wall, a cylindrical projection or a CAVE rather than a RWB or a ReachIn display system. An adequate combination of input devices and output devices has to be found with respect to the user's task and data set in use. Thus input and output device combination is directly derivable from the User+Need space as all needs and requirements are already defined there. An example of determining the User+Need Space for the example of 3.2 is given in section 3.8.

The Work Mode is also determined by the users' tasks. According to this thesis' focus, the work mode is mainly determined by the user, sharing of data model and collaboration needed. In particular, different modes of work relevant to this thesis are:

- stand-alone, autonomously and data sets are locally uploaded
- stand-alone, autonomously and data sets are remotely uploaded
- stand-alone, collaboratively and data sets are locally uploaded
- stand-alone, collaboratively and data sets are remotely uploaded
- distributed, collaboratively and data sets are provided by one of the sites, or by a remote (external) data server

The first two items describe the possibility of working alone where data sets are locally available or must be downloaded remotely, for instance from a simulation loop. No collaborative working is enabled at all in these two cases. The third and the fourth item describe collaborative working together using the same display system. Users are physically at the same place. The data sets are available locally again or have to be downloaded from a remote data server.
The last item is the more general one where at least two different sites work together. Now the shared data sets can either be provided by one or more members of the session or by an external data server. The work mode itself is important to determine the interaction metaphors described in sections 3.7.4 and 3.7.5.

3.5 Representation Components

Representation Components denote a very important part of Virtual Environments. They determine how the visual parts in the application are represented.

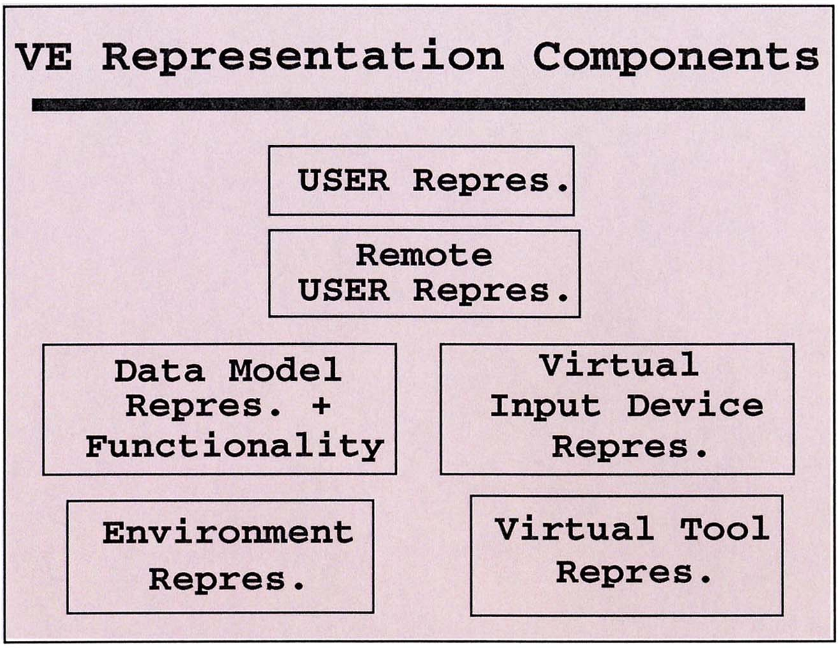

VE Representation Components
USER Repres.
Remote USER Repres.
Data Model Repres. + Functionality
Virtual Input Device Repres.
Environment Repres.
Virtual Tool Repres.

Figure 3.5: The VE representation components determine how the visual parts in the Virtual and Collaborative Virtual Environment are represented.

The components are (see Figure 3.5):

- User Representation
- Remote User Representation
- Data Model Representation and Functionality
- Environment Representation
- Virtual Input Device Representation
- Virtual Tool Representation

As shown in Figure 3.5 all components belong to a group although the *User Representation* has a special function. Most rear projection-based Virtual Environments do not need an explicit user representation in contrast to HMDs, where the user is typically represented by a hand or a whole body like for example in *Third Person Shooting* games.

The *remote user representation* represents the participating user or group of users from the other site. The aim of this representation form is to let the user or the group to appear present from a remote Virtual Environment. Therefore the factor of realism needs to be considered when designing an application. However, this depends on the task of the users. Sometimes more abstract user representations fit the requirements [80]. Well-established methods of user representation are avatars and real time video textures. Research on avatars has produced from very abstract to very detailed human representations that include realistic visual and physical models [23]. Research on using real-time video is using stereoscopic or mono video and different texture mapping and image manipulation techniques [67]. The advantages of video conferencing are the high realism and the ease in handling of the video texture in terms of positioning and scaling. The disadvantages are the transfer of video streams and network requirements. Also the alignment (matchmoving) of the texture with the virtual tool and input device representations needs to be considered.

The *environment representation* reflects the ambience of the users' physical environment. These representations can, for example, be an operation theater for surgeons, a lecture room for a professor and the students or a laboratory for a group of engineers. Environment representations are able to increase the feeling of immersion, as users might feel more comfortable in their natural working environment than in an abstract one. Especially when using Virtual Environments for training purposes environment representations facilitate to transfer the learned in order to repeat it in real world.

The *data model representation* is the data set of interest. Depending on the application these data sets can be, for example, a human body reconstructed from MR and CT recordings and a saw and drill for the surgeons, the car model with seats and crash test dummies for the engineers or the set of molecules for the chemist. Data sets of interest can either be abstract models or realistic synthetic models reconstructed from scanner data for example. The best representation format is determined by the possibilities of scientific visualization and the requirements of the user's task. When interacting with the data the different possibilities that denote its functionality have to be represented (see Figure 3.6).
Applications for experts exploit the real-world knowledge of the user which intuitively leads to the right way of interacting with the data, whereas in Virtual Environments for training purposes, functionality needs to be represented in an easy to perceive way. There exist two main ways in VEs to show functionality to the user [47, 68]. One is to offer static menus which pack the whole set of operations that are applicable to the data sets. It is obvious that there are plenty of different possibilities to visualize these menus. When choosing this type of functionality representation, the application designer and the pro-

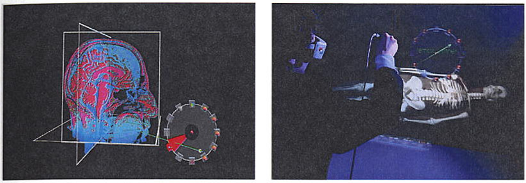


Figure 3.6: The ring menu belongs to one special data set. When a user asks the data set for its functionality this menu appears and shows the operations that are applicable to it. The menu is attached the user's hand position. It only follows the translation of the user's hand whereas the rotation of the user's wrist is used to intersect the "cake pieces" with the pick ray. Thus selection of operations is possible.

grammer have to find the most suitable way. Although a lot of work has been done in the area of HCI there is still a lack of guidelines for the specific set of applications which are the focus of this thesis. Problems which occur with those static menus are related to the limited interaction space of the displays systems and the uncomfortable usage when clicking through menu levels (see Figure 3.6). It has been proven that it is a much better strategy to ask the data set for its functionality rather than to try and address a certain functionality with a selected tool. Then the data set's answer can be displayed as a menu again which is fixed positioned somewhere in the VE or attached to the user's gaze or hand [49, 71].

The *virtual input device representations* reflect the active physical input device the user has chosen. Examples of these representations are virtual colored rays when using the stylus or multiple button devices. These rays enable the user to see where the physical input device or the hand points to and they facilitate the selection process (see Figure 3.7).
The *virtual tool representations* reflect the active tool a user has chosen. These representations could be 3D icons which are connected to the physical input device in use. Thus they follow the movements of the physical input devices or hands. With the help of these tool representations the user is aware of the possibilities of the active tools at any time (see Figure 3.7).

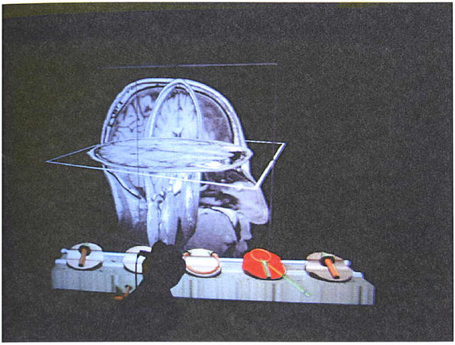


Figure 3.7: The toolbar does not belong to one special data set. It groups generic operations that are applicable to all kinds of data. The toolbar consists of buttons and virtual tool representations like 3D icons. When a button is pressed the icon is attached to the user's input device and replaces the virtual input device representation which is a pick ray in this example. The icon disappears from the toolbar highlighting that the tool is active.

3.6 Awareness-Action-Feedback Loops (AAF)

The Application+Interaction space describes how users interact, with each other and the data set of interest, collaboratively in the Virtual Environment. In order to find the best interaction first the low-level makeup of interaction has to be understood. Therefore interaction tasks have to be narrowed down and interaction templates have to be found which can be combined to form more complex interactions.

Awareness-Action-Feedback loops denote such interaction templates (Figure 3.8). These AAF loops provide the possibility to understand and analyze very tiny steps in interactions.

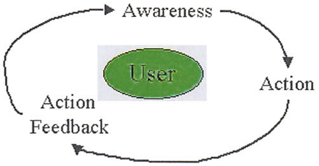


Figure 3.8: Awareness-Action-Feedback loops are interaction templates with which it is possible to analyse the low-level makeup of interaction.

3.6.1 Autonomous AAF Loop

Before explaining complex collaborative interactions it is started with autonomous interaction (see Figure 3.9). The autonomous AAF loop is divided

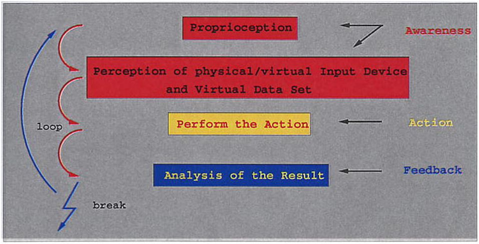


Figure 3.9: Diagram of the autonomous Awareness-Action-Feedback Loop.

into four blocks. The first two blocks belong to the awareness phase where the user starts with proprioception as defined by Boff et al. in [7]. The proprioception allows the user to become aware where it stands and looks to, the position and orientation of body parts like arms, hands and fingers and everything that is needed for interaction. It is the perception of the user in relation to the environment. The next step is to be aware of the physical input devices held in the users' hands and the virtual tool representations connected to them. The position and orientation of the virtual data set is perceived in this phase as well. After the user is aware of the representation components and herself the action phase follows. This action can simply be to move the hand holding the physical input device.

Upon completion of the action phase the feedback phase starts. This feedback is an action feedback without which it would not be possible to analyze the result of the action. In this case the user perceives the movement of the virtual tool representations as s/he moves the input device together with the hand. After the perception of the status of the situation the user decides whether the task is completed and therefore wants to break the loop or whether the task is not completed yet and therefore prepares for the next action within the same loop.

The AAF loop is exemplified for the real scenario of a carpenter who wants to pound a nail into a piece of wood with a hammer. The steps of the AAF loop are:

1. **Proprioception** → *Awareness*
   Where am I ? Where do I look at ? Where are my hands and fingers ?

2. **Perception of the physical/virtual input device and data set** → *Awareness*
   Where do I hold the stylus ? Is the hammer connected to my hand ? Where is the piece of wood ?

3. **Perform the action** → *Action*
   Interaction of human body (hands, fingers etc.) and physical input device. Position the nail on the wood and position the hammer.

4. **Result Analysis** → *Feedback*
   Perceiving the status of the situation. Perception of position, orientation and status of the virtual data and input device. (e.g. Did the data set allow the operation ? Is the nail positioned correctly ? Is the hammer in place and ready to pound ?)
   Depending on the status return to step 1. and proceed or break the loop (e.g. I am not ready yet so proceed with pounding the nail.)

5. **Repetition of steps 1/2/3/4** until the task is completed.

3.6.2 Collaborative AAF Loop

Collaborative Awareness-Action-Feedback loops are of the same structure as the autonomous AAF loops (see Figure 3.10).

The main difference between them is that the collaborative AAF loop needs to additionally address collaborative requirements that are necessary when working in a team. Again the collaborative AAF loop starts with the proprioception

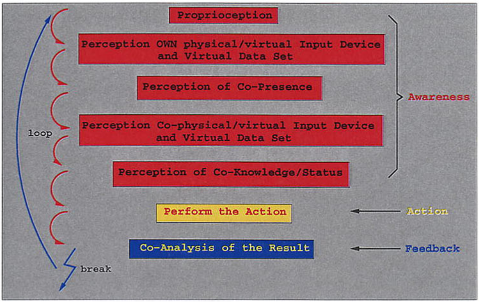

Proprioception
Perception OWN physical/virtual Input Device and Virtual Data Set
Perception of Co-Presence
Perception Co-physical/virtual Input Device and Virtual Data Set
Perception of Co-Knowledge/Status
Perform the Action
Co-Analysis of the Result
Awareness
Action
Feedback
loop
break

Figure 3.10: Diagram of the collaborative Awareness-Action-Feedback Loop.

block and the perception of the own physical input devices and the virtual tool representations. After this but still in the awareness phase the user perceives the co-presence. It is comparable to proprioception but now information about the remote partner is queried: Where is my partner, where does he look at, where are his hands, fingers etc. Similar is the perception of the physical input device and the virtual tool representations together with the virtual data set. An interesting component represents the perception of co-knowledge and co-status. It is often not sufficient to know where you and your partner are located and where the object and the tools are when working in a team. It was possible to find out that knowing that your partner is aware of you is one of the most important steps in the awareness phase (see chapter 6). To know that your partner is aware of what you are intending to do and what you want to achieve with your action is essential for team work. Everything that supports this type of awareness increases the amount of collaboration. While perceiving the co-status the users check the situation. The users can confirm this status check, by voice or with the help of a gesture like the "thumbs up". The action and the feedback phase are equal to the ones of the autonomous AAF loop.
In order to apply the collaborative AAF loop to a real scenario assume two carpenters who again want to pound a nail into a piece of wood with a hammer. One carpenter holds and positions the nail on a piece of wood and the other carpenter pounds the nail with a hammer. Then it is possible to describe the whole interaction task from the carpenter's point of view who holds the

hammer.

1. **Proprioception** → *Awareness*
   Where am I ? Where do I look at ?
   Where are my hands, my fingers ?
   The same as before (see AAF loop).

2. **Perception of the physical/virtual input device and data set** → *Awareness*
   Where do I hold the stylus ?
   Is the hammer connected to my hand ?
   Where is the piece of wood ?
   The same as before (see AAF loop).

3. **Perception of co-presence** → *Awareness*
   Where is my partner ? Where are his hands and fingers ? Where does my partner look at ?

4. **Perception of co-physical/co-virtual input device and data set** → *Awareness*
   Where does my partner hold the nail and the wood ? How is the relationship between nail and wood ?

5. **Perception of co-knowledge and co-status** → *Awareness*
   Is my partner aware of me ? Does he know where I am, where I am looking at and where I hold the hammer ? Does he know what I am doing and what I want to do ? Is everything ready now ? Confirmation of the status check by voice or "thumbs up".

6. **Perform the action** → *Action*
   Interaction of human body (hands, fingers etc.) and physical input device. Position the nail on the wood and position the hammer. The same as before (see AAF loop).

7. **Result Analysis** → *Feedback*
   Perceiving the status of the situation. Perception of position, orientation and status of the virtual data and input device. (e.g. Did the data set allow the operation ? Is the nail positioned correctly ? Is the hammer in place and ready to pound ?)
   Depending on the status return to step 1. and proceed or break the loop (e.g. We are not ready yet so proceed with pounding the nail.) The same as before (see AAF loop).

8. The **steps 1. to 7.** are repeated until the task is completed.

3.7 Operations, Metaphors, Interaction Techniques

Awareness-Action-Feedback loops like the ones shown in Figures 3.9 and 3.10 are templates. With the help of operations, metaphors and interaction techniques it is now possible to give those templates a "face". This means that depending on the user's subtask the appropriate operations, metaphors and interaction techniques have to be chosen for each action.

Operations defined in the taxonomy provide the means for supporting manipulation of virtual data and shared manipulation between remote participants. They describe what can be done with the virtual data in terms of how the data is explored. They can be data independent (i.e. basic operations such as selecting), or data dependent (i.e. slice through a 3D data volume). Refereing to the definition of interaction techniques and interaction tasks in section 2.1 of chapter 2 operations denote the action itself rather than the techniques that are used to put the operation into practice.
In this work three categories of operations are identified, generic operations, content specific operations and collaborative operations.

3.7.1 Generic Operations

Generic operations are used to manipulate virtual data sets of different kinds (see Figure 3.7). There exist a lot of generic operations and to mention only a few: *translation, rotation, zooming, dragging, pushing, deleting, grabbing, highlighting, selecting.*

In this approach selecting data is an even more fundamental generic operation since selection is the basis for all kinds of generic operations listed above.

3.7.2 Content Specific Operations

The other category of essential operations is content specific (see Figure 3.6). The virtual data sets determine additional operations that are meaningful depending on their features and nature. They are categorized as follows:

- change the mode of visualization
- change the appearance of the data (i.e. color and material attributes including textures, highlighting parts)

- change the geometric shape of the data (i.e. deformation, cutting, clipping, slicing)
- change the relationship of parts of the data (i.e. between different data sets or parts of the same)
- start/stop/pause of sequential data (i.e. videos, simulation loops, animations)
- reading and editing, descriptive text
- sonification of actions, events or text
- connecting/disconnecting with remote data servers or clients
- others

3.7.3 Collaborative Operations

All of the generic and content specific operations for the autonomous user mode can also be used in a collaborative session. Again data sets have to be translated, rotated and zoomed in or out. However, the operations need to be extended to include shared manipulation. Furthermore, additional operations are needed to establish and control a collaborative work session.

In the first category there are operations for:

- sharing of virtual data sets
- sharing different views on data sets
- sharing of operations

The data sets as well as the operations need to be distributed. A global operation box, similar to a tool-box, is the basis for sharing common operations in addition to the local operations at each site. Sharing different views of the data is important, for example when users like to concentrate on different aspects of the data set requiring different visualization modes.
In the second category there are operations for:

- establishing a session
- controlling positioning
- controlling conversation between participants

- terminating a session

More specifically, calling, hanging up, muting a video and audio connection at any time are generic operations dealing with the audio/video communication of remote users. Also switching between different remote partners or seeing and hearing them all at once is possible. Positioning of the remote participants' video representation in one's working environment allows control over the team's position and supports team dynamics.

3.7.4 Autonomous and Content Specific Metaphors

Metaphors for interaction and collaboration make use of everyday interaction and collaboration paradigms to provide intuitive ways of interaction in Virtual Environments (i.e the metaphor of working around a table). In this thesis three categories of metaphors are defined.
Stand-alone Metaphors such as walk, fly and teleport, directly use or extent real-life paradigms to allow navigation through a Virtual Environment. *Content specific metaphors* that allow the user to focus on an interesting part of the data set, look closer, hear/touch interesting subparts, as well as additional ones like play video/TV, query information, can also be adapted from real-life paradigms.
However, there can be more than one way of combining operations to implement a metaphor. For example the teleport metaphor can make use of the zoom operation in order to scale the data set and let it appear larger to the user. In addition, it is also possible to apply a translation operation in order to either change the user's position to be closer to the teleporting point, or to move the data closer to the user. Depending on the application and the type of virtual data, one metaphor might be more intuitive than others. It is very useful to scale an object when observing interesting parts that retain the view on the surroundings. Moving closer to the object of interest could be useful for further operating on it. The metaphor that corresponds to Newton's law of action and reaction can either use the virtual data set or the user as point of reference. Concerning the effect, it makes no semantic difference whether the user moves around the object (*ego-centric manipulation*) or the object rotates around its axis and the user's point of view is stationary (*exo-centric manipulation*). Both implementations, ego-centric and exo-centric have their merits. The approach is to make the different metaphors transparent to the user and allow to choose the metaphor best suited for the task. This shows that generic, as well as content specific operations, can be used to implement metaphors.

3.7.5 Collaborative Metaphors

The *collaborative metaphors* are categorized in this work as follows:

- visual and verbal communication between users
- sharing viewpoint of participants
- virtual/verbal/tactile manipulation and sharing of data sets

Metaphors for visual and verbal communication include, working around a table, working next to each other, working on different parts of the data set, walkie-talkie and turn-taking verbal communication. The verbal communication metaphors, especially when using speech recognition, distinguish between voice commands and audio communication for talking to other participants. The user may want to give commands to the computer and to share them so that the remote site is aware of these commands. Using the walkie-talkie metaphor the remote site cannot disturb or interrupt the local user giving verbal commands to the system. Using the turn-taking metaphor the computer does not have to listen during the user's verbal communication. Important when using verbal communication is to simulate the real-life situation where the voice of a participant closer to the user is louder. This enhances the user's perception of presence and co-presence. Attaching the audio stream to the position of the video-avatar of the remote participant using localized sound sources is a possible implementation of this real-life paradigm.

Metaphors for sharing viewpoints include:

- sharing each other's viewpoint (look over the other's shoulder)
- same viewpoint (look through one's eyes)
- mirrored viewpoint (opposite side of table situation)

Finally, metaphors used for collaborative manipulation need to provide the possibility that all participating users manipulate a shared object at the same time using the same or different tools. This possibility is provided by the tug-of-war metaphor. An alternative metaphor that avoids deadlock situations is the tug-of-war with dead end metaphor. Each site receives two versions of the same shared object. One can be manipulated only by the local user and the other only by the remote user. This avoids conflicts but might require a bigger virtual space which can be limiting. Very basic locking mechanisms can also be used in order to avoid deadlock situations. Appropriate feedback types that indicate which user locks the data are required then.

3.7.6 Interaction Techniques

In contrast to the metaphors the interaction techniques determine how to support and implement the different types of operations. Again there are plenty of different ways to do this [49]. In order to make interaction in Virtual Environments richer and more intuitive, techniques have to be provided which use more than only one of our senses at the same time. Some interaction techniques that have proved to be quite adequate in terms of intuition as described in chapter 2 are:

- speech recognition
- tactile and force feedback
- menus
- virtual pick-ray
- toolbar/toolbox
- body-centered interaction
- gesture recognition
- olfactory interaction

Physical mnemonics and other senses have been successfully used to store and recall information relative to the body using hands, eyes, or even the whole body[71]. Depending on the available media and interaction devices, the defined operations can be implemented in different ways. For example, the selection operation can be implemented by recognizing simple voice commands or by the use of an interaction device (i.e. stylus) and a virtual pick ray. The perceived quality of interaction depends extremely on the interaction devices and their use. In order to name a few: *tracked shutter glasses, 6DOF pen-like stylus, multiple button tools with location sensors, data-gloves, the cubic mouse[41], tactile and force feedback devices, such as the Phantom from Sensable, and joysticks with location sensors, and many more described in chapter 2.*

The developer has to carefully select the most appropriate ones according to the VE. Recommendable techniques are those which enable the user to concentrate on the task and not on steering through menus and toolboxes.
Collaborative interaction techniques are the same as in the autonomous user mode. There is no need to develop new techniques in order to perform collaborative tasks. When being shared, menus, pick rays, voice recognition and other interaction techniques are used the same way as in the single user mode. A detailed taxonomy of interaction techniques can also be found in [12].

3.8 Example CVE design

In this last section the simulation of an example CVE design is made. With this example the reader will be able to understand the make-up and the process of CVE design making use of the VE/CVE design model developed from the taxonomy of influence factors at the beginning of this chapter.
It is possible to see how all representation components, work mode, operations, metaphors and interaction techniques can be chosen. For doing this the User Task description and the User Task analysis from section 3.2 is used. Thus the determination of a User+Need Space and the Application+Interaction Space has to be seen with respect to this UTD and UTA. Each item shows the corresponding representation briefly. The substantiation below each item indicates the choice for the special implementation.

User Representation

- no representation
  - reason: When using rear-projection systems no representation form for the user itself is necessary in contrast to using HMDs (see *output devices* and section 3.5 too)
- video image of the partner
  - reason: Using the video texture in rear-projection systems is a useful representation form for the remote partner (see section 3.5 too)
- Avatar
  - reason: Also a possible representation form. But as it is very important to see the movements of the partner working on the roof. So if using an avatar then it must include joints, muscles, motion etc. for the simulation of this task.

Representation and Functionality of the Data Sets

- virtual models of the wooden laths including the corresponding material properties of wood (ex. "woodiness")
- virtual models of the box with nails including the corresponding material properties of iron (ex. "steeliness" in order to pound them into wood)

Representation of the Environment

- virtual model of the roof (the viewpoints of the users have to follow gravity and to underly a collision detection with the roof)
- virtual models of the landscape, the sky, surrounding houses and other roofs which provide a feeling of height, depth and wideness (eventually acoustic or even physical wind representations)

Representation of the Tools

- virtual models of the hammers (including steeliness and weight if using force feedback systems)
- virtual models of the pair of pliers (see above)

Representation of the Input Devices

- virtual models of the users hands
  - note: Only useful when using data or pinch gloves as input devices
- virtual pick rays
  - note: Useful when using a stylus (pen-like input device)

Work Mode

- stand-alone, collaboratively and data sets are locally or remotely uploaded (for the case the two users are at the same site)
- distributed, collaboratively and data sets are provided by one of the sites, or by a remote (external) data server (for the case the two users are at different sites)

Input Devices - Output Device Combination

- stylus (practical)
  - reason : Easy to handle, suitable for both hands
- multiple button tool (practical)
  - reason : Easy to handle, suitable for both hands, due to the buttons more functional but here not really needed due to the simple task

- pinch or data glove (not very practical)
  - reason : Good representation of the using hand and thus direct manipulation is possible. But it can only be used with one hand. Further it is uncomfortable to use when trying to pass it to the partner unless the partner works remotely.
- video camera
- microphone (necessary when working distributed, collaboratively)
- Responsive Workbench (not practical)
  - reason : Handling the data set and working with the tools can be displayed. But working on a roof and having the collision detection with it is impossible to display within the limited view frustum.
- CAVE (practical)
  - reason : Handling the data set and working with the tools can be displayed. In addition to that working on a roof and having the collision detection with it is possible to display within the larger view frustum. Even when working stand-alone, collaboratively.
- Wall (not practical)
  - reason : As the display consists of one front screen and has no "floor" projection screen orthogonal operations are not possible without rotating the whole scene. This working metaphor does not exploit real-work-knowledge and thus is very unnatural (see section 3.7.4).
- cylindrical projection (not practical)
  - reason : The reasons are similar to the ones given for the wall display. One difference to the wall is that this display can provide a greater feeling of immersion through its surrounding character but the ground projection is also missing.
- speakers or headphones (practical and necessary when working distributed, collaboratively)

Generic Operations

- **select/grasp** - (either the nail or the hammer or the wood)

  Action

  * points with the virtual pick ray to the data set and presses a button of the stylus or multiple button tool
  * points with the finger to the data set and snips with the thumb and the middle finger when using pinch gloves

- **translate** - (e.g. in order to position the nail on the wood)

  Action

  * the user 'selects'/'grasps' the nail or the wood and moves the hand together with the selected object (this action is restricted because of the limited interaction space on top of the roof)
  * the user selects a special translate tool and applies this to the selected data set
  * the user selects a special drag tool which is a combination of a translate and rotate tool and applies this to the selected data set

- **rotate** - (e.g. in order to position the wooden laths to each other or on the roof)

  Action

  * the user 'selects'/'grasps' an end of the wooden laths and moves the hands (remember the laths are heavy $\longrightarrow$ so two-handed interaction is required)
  * the user selects a special rotate tool and applies this to the data set
  * the user selects a special drag tool and applies this to the selected data set

- **zoom** - (e.g. in order to get a closer and better view on the data set)

  Action

  * the user 'selects' and 'translates' the data set and moves the hand closer to the eyes
  * the user changes its position (goes closer to the data set)
  * the user selects a special zoom tool and applies this to the data set

Content Specific Operations

- **change the geometric form of the data sets** - (e.g. when the nail goes inside the wood)

  Action

  * the user 'selects'/'grasps' the hammer, moves the hand and pounds on top of the nail

- **change of the view point** - (e.g. for verification purposes)

  Action

  * the user presses a button on the stylus or multiple button tool and 'rotates' the whole scene
  * the user presses a button on the stylus or multiple button tool and 'rotates' its viewpoint as long as the button is pressed

- **undo function** - (e.g. for pulling the nail out of the wood)

  Action

  * the user 'selects'/'grasps' a pair of pliers, selects the nail and pulls the nail while moving the hand
  * the user selects a special undo tool and applies this to the data set

Metaphors

- for the position of the partners:
  - *eye-to-eye contact, data set between the partners*
- for the communication between the partners:
  - *turn-taking conversation*
- for the rotation operation:
  - *rotate the data sets*
  - *walk around it but never rotate the whole scene as the users stand on the roof*
- for the zoom operation:
  - *change user's position (go closer to the object)*
  - *scale the data set itself*

- for the collaborative manipulation of the data sets:
  - *tug-of-war* : This metaphor allows both users to apply operations on the data set at the same time. No locking mechanism is implemented. The advantage is that the first user can hold the wooden laths with one hand and position the nail on top of them whereas the second user holds the wood with one hand and pounds the nail with the hammer in the other hand. The drawback is that both users exactly have to know what the other one is doing as uncoordinated interaction will result in a real tug-of-war situation.
  - *look through someone's eyes* : this metaphor enables the users to slip into the partners position and to have a look from the other side onto the data set. This metaphor is very useful when working opposite each other within a limited virtual work space.

Interaction Techniques

- tactile/force feedback : for the perception of the data sets and the tools specific weight, for tactile feedback when holding a hammer and a nail, force feedback when pounding nails
- menus : for the generic operations: zoom, rotate, translate
- virtual pick ray : for the selection with the stylus input device and in order to apply the operation
- sonification : for the acoustic feedback of the hammer blow
- Gesture recognition :

Types of Feedback

- changes of the data sets (highlight the nail if hit correctly, highlight the wood if the nail is positioned correctly, nail goes into the wood - wooden laths cannot be moved freely)
- acoustic feedback (hammer blow, 'hit/not hit'-Sound))

3.9 Conclusions

This chapter introduced a complex, theoretical interaction framework for Collaborative Virtual Environments. With the help of this framework Virtual

Environment designers and programmers are able to analyze user tasks and interaction cycles. The results of the analysis, then, determine a User+Need Space (UNS) which itself describes the appearance of the Collaborative Virtual Environment according to the visual representation. Performing the mapping of the UNS to the CVE, representation components are elaborated. Then together with the analysis of Awareness-Action-Feedback loops for autonomous and collaborative interaction cycles the desired Application+Interaction Space is determined.

The next chapter 4 determines an Application+Interaction Space for a predescribed two user task scenario. The focus there is on the use of two networked Responsive Workbenches. However, interface and application specific information concerning the coupling of a Responsive Workbench with a CAVE is also provided. In chapter 6 the developed applications are evaluated assessing usability and collaborative awareness.