[WSSa94b] *Marcus Wieland, Ralf Steinmetz, Peter Sander;* **Remote Camera Control: Requirements Concepts and Implementations**; Technical Report 43.9405, IBM European Networking Center Heidelberg, Germany, 1994, Auszug erschienen als [WSSa94a].

Remote Camera Control: Requirements, Concepts and Implementation

Marcus Wieland, Ralf Steinmetz,
Peter Sander

Technical Report 43.9405

IBM European Networking Center
Vangerowstraße 18 • 69115 Heidelberg • Germany

Phone: +49-6221-59-3000 • Fax: +49-6221-59-3400

{willy, steinmet, sander} @ dhdibmip.bitnet

Abstract:
As an application for the *Heidelberg Transport System* (HeiTS) for multimedia communication, the IBM European Networking Center (ENC) has developed HeiCam, the *Heidelberg Remote Camera Control*. The goal of HeiCam is to exploit digital multimedia communication in connection with control of a camera in a surveillance or CIM environment. In such a set-up, the camera is attached to a computer which is driven by an application running on a remote machine. The distribution of the audio and video data is performed by an audio/video subsystem based on HeiTS and controlled by HeiCam. HeiCam runs in a distributed multimedia environment on both, OS/2 and AIX, platforms. This paper outlines the major goals, design items and implementation experiences.

Keywords:
Distributed multimedia systems, live video, remote camera control, human interface, surveillance

1 Introduction

Most publications on multimedia issues address basic technology or system software aspects (video server implementation, communication protocols, etc.). Some papers discuss networked applications with the focus on multimedia conferencing [Alte93], multimedia and hypermedia documents [Niel90] and video on demand [Meye91] with client and server systems. We look into details of networked applications, a remote camera control application.

We designed and implemented the HeiCam application to exploit the advantages of existing computing and communication for the purpose of a remote camera control. Remote camera control facilities are mainly used to observe buildings or restricted areas, especially where danger to life exists or employment is unprofitable, as in chemical plants or nuclear power stations. First, let us describe in detail how a remote camera control works in a networked environment.

The camera is attached to a computer, the "camera server", via an electrical interface, for example a serial RS232C interface. Control commands like "move", "zoom" and "focus" are sent via this interface from the computer to the camera. The originator of the commands is an application running on a remote computer, the "camera client". Both computers, client and server, are interconnected by a computer network, for example a Token Ring or Ethernet LAN. Audio and video data is continuously captured, coded and compressed at the server. This data is transmitted over the network to the client. At this remote location the audio/video signal is displayed in the window of the client application. In these set-ups the camera attached to the camera server can be controlled remotely from one of the clients. The user controls horizontal and vertical movements by clicking with the mouse on appropriate buttons or on some part of the displayed video image itself. All other control functions of the camera are also reflected appropriately at the user interface at the client workstation.

We learned that the visual control of critical production processes would be easier if video pictures from the process could be displayed together with corresponding measurements data on a single display. Control consoles for production supervision have always been developed with the goal of integrated solutions: The control personnel should have all the relevant information at their disposal in a compact and precise form. Up to now it has not been possible to integrate video using the same facilities. Consequently separate displays had to be installed in control centers. There are advantages to retrofitting video monitoring into existing control consoles. Video pictures can be shown in one window while other windows display measurements and alarm indicators along with schematics of the production process. With an intelligent presentation system, it is possible to display only relevant video information at a certain point of time or at a specific state of the industrial process to be supervised. This brings us another step closer towards a complete integration of computer integrated manufacturing.

From a technical perspective, a camera installed in the production area must be controlled from a remote workstation. The physical distance between camera and the controlling workstation is of no concern. The control commands for the camera as well as the live video (and perhaps also audio) data are transmitted over the same digital network. Other scenarios based on the same live video technology include the use of cameras in security systems.

In our multimedia work we experienced that most of the applications heavily depend on some specific multimedia operating system extensions, some well defined compression schemes or even some operating systems and communication networks. Therefore, with HeiCam we decided to overcome these shortcomings and make this application available for different computing environments. We have HeiCam on PS/2 with OS/2 [Sand92, Lier93] and on RS/6000 with AIX [Wiel93].

This opens a wide market and the access to a large number of PS/2s and workstations. If a camera is attached to a PS/2, there is no difference in controlling the camera from a workstation or another PS/2. The same video data, similar user interfaces and same features are included in both, OS/2 and AIX, versions which interwork across the platforms.

The remainder of this text is divided into four sections. Section 2 outlines the environment and constraints of the HeiCam application, whereas the main user requirements are presented in Section 3. Section 4 describes the most crucial issues of our implementations. Section 5 concludes summarizing the main features, our experiences and future issues to work on.

2 Environment and Constraints

HeiCam has been developed in various versions over the last 3 years. Based on the initial prototypes a user requirement study was performed. We wanted to find out the specific user demands of what and how to remotely control a camera. In the following we outline these issues and describe how they are reflected in HeiCam.

Network Environment

A main motivation for the use of this application is the desired reduction of expenses in association with the installation of any distributed remote camera control. The existing infrastructures, namely computer networks and the attached workstations should be used. It should be avoided to invest in dedicated equipment such as analog cabling for the transmission of video signals.

During the last years, there is a discernible trend from stand-alone workstations and PCs to networked systems. The integration of workstations into a networked environment offers the opportunity to share expensive peripheral devices such as laser printers, scanners and in principle also audiovisual equipment like cameras. The other principal usage is the exchange of data. Conventional LANs provide fast bulk data transfer in a fair mode without real-time demands. To a certain extent these networks can also function as "shared medium" for the exchange of audio/video data.

In such an environment a workstation can act as client and/or server. The HeiCam application incorporates this client/server paradigm: Several servers and several clients are attached to the same network. The server provides video data to the clients using multicast facilities of the communication protocols. Any client is able to control every camera attached to any server. Servers have to be equipped with special video hardware for framegrabbing, digitizing and compressing the desired video signal. Depending on the employed compression scheme the client might also be equipped with hardware to speed-up the decompression and display process.

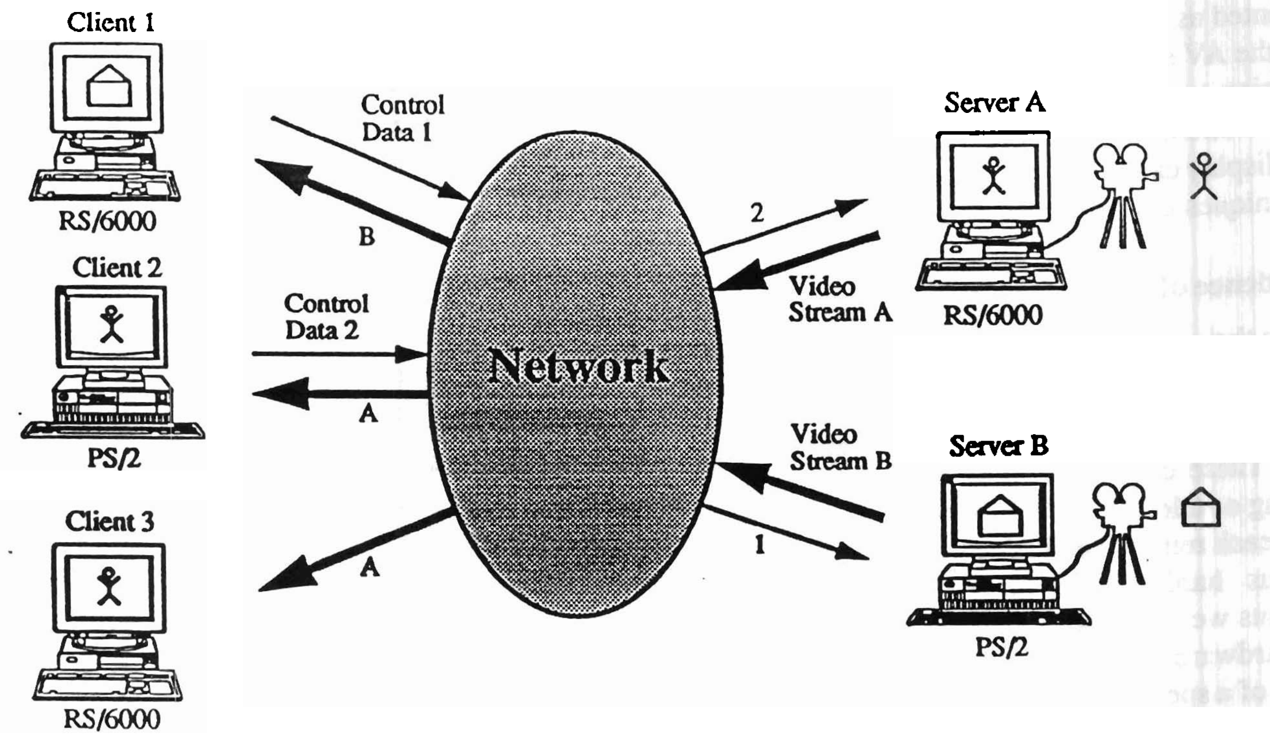

Figure 1: Network Environment

Figure 1 shows a typical scenario of the HeiCam application and symbolizes the network environment: Server A and Server B feed their video data into the network. The clients can choose which video streams they want to attach to. Client 1 "consumes" the "video stream A" and controls the respective camera with the "control data flow 1". Client 2 displays the video data of stream B and sends the control data flow 2. The HeiCam application at client 3 can display either video stream.

Independence of a Specific Audio and Video Format

As outlined in [Stei94a, Stei94b] there already exist many compression techniques. Most of them are already used in today's products while other methods are still undergoing development or are only partly realized. The most important compression techniques are JPEG (for single pictures), H.261 (for video), MPEG (for video and audio) and proprietary developments like DVI, Intel's Indeo, Microsoft's Video for Windows, IBM's Ultimotion, Apple's QuickTime or DigiCipher II developed by General Instruments Corp. and AT&T.

JPEG, H.261, MPEG and DVI all have slightly different technology and application areas. Most of the used algorithms are very similar but, not the same. The technical quality as well as the availability on the market determine the techniques that will be used in future multimedia-systems. This will lead to a "cooperation" and "convergence" of the techniques. For instance, a future multimedia computer could generate still images in JPEG, use H.261 for a video conference and MPEG-2 as well as DVI PLV for retrieval of stored multimedia information.

This large number of compression schemes and their ongoing developments demands for adaptability to various formats and implementations. The processing of video data is not specific to HeiCam, moreover it is basic to all kind of multimedia application. It is typically implemented as an extension to the operating system with local functions only. HeiCam interfaces to the AV subsystem. The use of this separate subsystem allows the integration of further compression techniques and capture/display hardware adapters outside the scope of the HeiCam application. The AV subsystem requires new device drivers for each new technology and capture/display cards. Therefore, our HeiCam application itself is independent of any compression techniques or capture hardware.

Independence of a Specific Camera Type

Similar to the various formats of audio and video data and their integration into the audio/video subsystem, a closer look to the diverse camera hardware control interfaces is required.

The architecture of a distributed remote camera control should support a large amount of video cameras. These cameras differ considerably with respect to their intended use (for example monitoring or teleconferencing). We encounter considerable differences in the hardware interfaces for each remote camera control. HeiCam must allow for the integration of this set of heterogeneous hardware resources. Hence in analogy to well-known operating system components we introduce a novel interface software between control functions and the actual camera hardware: the "camera driver". A set of drivers for different cameras provides the independence of a specific camera type.

Further more, our approach guarantees

- the independence of the client application programs from a specific camera control device
- the reuse of the driver software for other camera control set-ups and other application programs
- the portability to other operating system environments.

Independence of the User Interface

A structure design of the HeiCam application into several functional elements eases not only the realization of the independence of compression techniques or specific camera hardware, it also allows the integration of different user interfaces. The user interface enables the user to control the camera from the client station. The additional definition of a programming interface to these control functions enables the development of text and graphics-oriented user interfaces.

This programming interface must contain all necessary functions to move, zoom and focus the camera, to store and retrieve camera positions. Also functions to start and stop the delivery of the audio and video data to the corresponding clients and functions to store and retrieve videos on demand are of interest.

Functional Blocks

A structured design of the remote camera control application results in the separation of the software into the following functional blocks.

- User interface
- Networked control functions for the camera control (including a camera driver at the server site) with the "HeiCam Control Protocol" (HCP)
- A networked audio/video subsystem with the "Source and Sink Control Protocol" (SSCP)

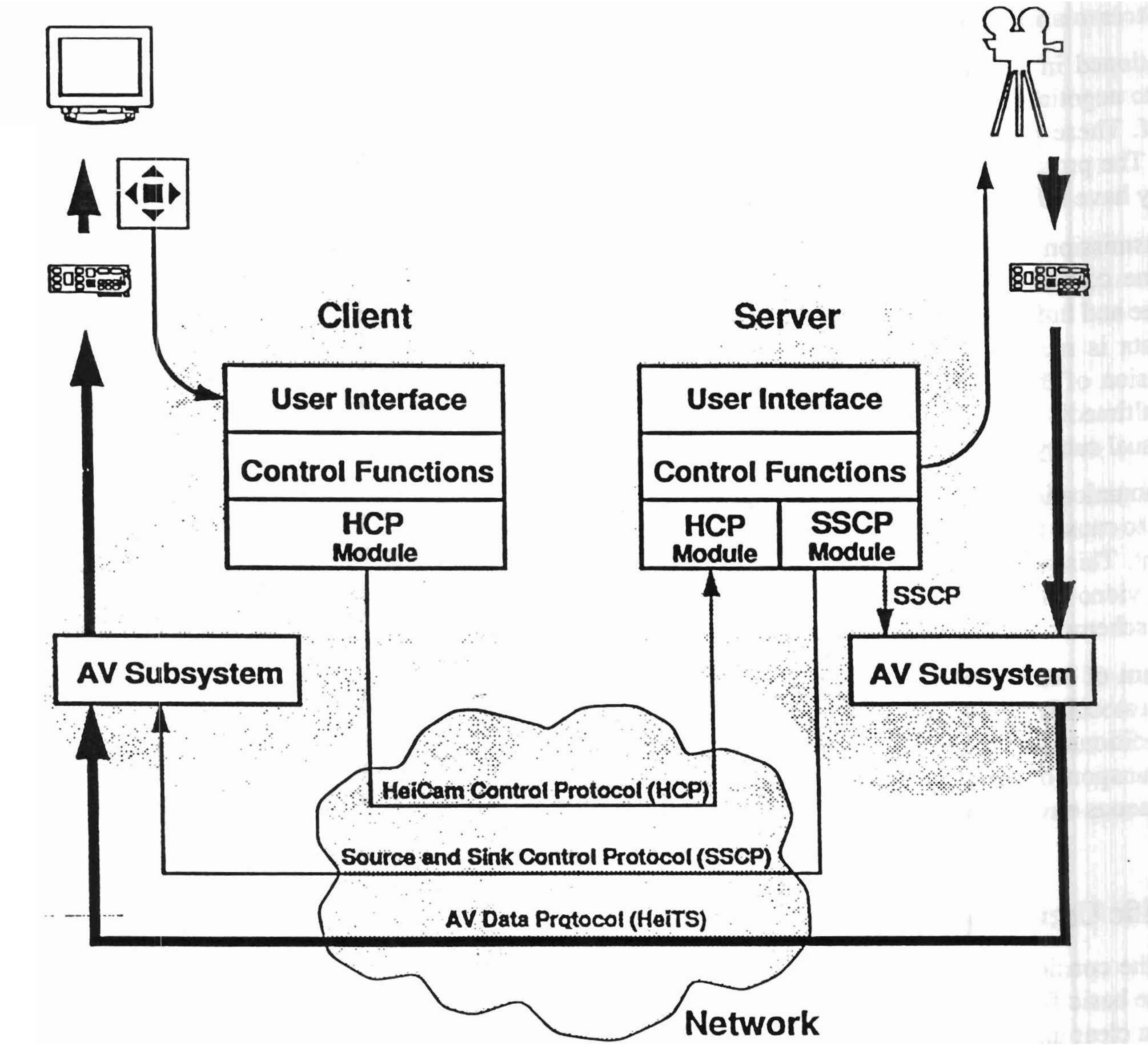

Figure 2: Data Stream and Control Stream Flows

The user interface enables the user to interact with the HeiCam application. It must provide a clear view of the available functions for the control of the camera which must be easy to use and easy to remember. The control functions module performs the commands issued by the user via the user interface. At the server site it processes the communication with the camera driver. Protocols for the communication between client and server and the communication between server software and the separated AV subsystem are needed. Therefore, interface modules to these protocols are required. The audio/video subsystem controls and manages the distribution of the audio and video data from a server to the several clients. It is developed separately, interfaced by HeiCam and entirely encapsulated. Note that also in our version with an object-oriented design and implementation we encounter the same functional partitioning as part of the class hierarchy [Lier93].

Video Stream and Control Protocols

As mentioned in the description of the functional blocks above, two control protocols are needed to negotiate and establish the actual video data transfer, and to control the remote camera itself. These protocols exist in addition to the protocol for the transmission of the video images. The protocols and the video stream are all transmitted over the digital network, however they have different requirements:

The transmission of audio and video data has to fulfil real-time requirements and must obey strict time constraints. A management of the involved resources, e.g., network bandwidth, CPU time and buffer space, has to be done. Therefore, a special transport system for audio and video data is needed. The Heidelberg Transport System (HeiTS) [Hehm92] processes the transmission of the AV data and ensures the needed resource management. In principle any other multimedia transport system can also be used as long as it interworks with the available audio visual subsystem.

The communication between the server and the AV subsystems of server and client requires a protocol to cause the AV subsystem to start or stop an audio/video transmission between server and client. This protocol is not strictly time-bound and it should provide for the selection of different video data rates, constant and variable bit rates as delivered by the respective compression schemes. It is known as SSCP, the Source and Sink Control Protocol [Alte93].

The stream of camera control data is not strictly time-bound, but it should be reliable. This control protocol is called HCP (HeiCam Control Protocol) and is based on RPC [Bloo92]. The use of traditional protocols, such as TCP/IP, as transport layer protocols of the RPC are suitable for the transport of the control commands and simplifies the exchange of these commands in heterogeneous environments.

3 Specific User Requirements

Besides the consideration of the above elaborated application environment and the introduction to the basic functional blocks, the most crucial factor of success in any multimedia application is a clear understanding of the specific user requirements.

Any user interacting with a remote control application is not only interested in having as many features and functions as possible; she/he demands for a fast response time to any given command and it should be very easy to use for a casual as well as a professional user. All these major points are the primary factor for the success or failure of HeiCam. All architecture and implementation issues must support and enable these user requirements.

Control Operations

In the design phase of a remote camera control, the question arises which control operations should be available for the user and how they are called. First of all, the user expects an intuitive way of handling the remote camera: Selecting and changing the shot the camera offers. The user needs, for better acceptance, hands-on movements and focus and other operations of the camera, as he would hold a camera right in his hands.

The HeiCam user interface offers therefore a set of functions a user will expect. There are

- Move, zoom and focus
- Video capture

- Defined camera positions
- Adjustments for image quality

Control Operation "Move"

The user of the remote camera control likes to control the camera in the same way that she/he would operate a local one. Using it manually, she/he is able to move around or keep it stationary (for example on a tripod) in one position with the capabilities to rotate. In practice, stationary installed video cameras often require remote control. In order "to move" the camera by a remote camera control system, the camera has to be mounted on a swivel-base and it may be driven by stepping motors which ideally should be moveable in each direction:

- Operations for spatial movement:
  - forward/backward/stop
  - left/right/stop
  - up/down/stop
- Operations for rotation
  - horizontal: rotate left/right/stop
  - vertical: rotate up/down/stop

Control Operation "Zoom"

Zoom operations allow to change the focus, they enlarge or diminishes the view of an image. It should be possible to use all available focus values the camera provides. The control of the camera objective by a remote control system is therefore a fundamental requirement. The following zoom operations should be available:

- Zoom out
- Zoom in
- Keep a fixed focus length

Control Operation "Focus"

To focus means to adjust the resolution of the video image for a given distance. This facility can be provided by autofocus and manual focus. Usually a camera offers both facilities. Operations the remote control system must provide are:

- Autofocus (on/out)
- Focus out
- Focus in
- Adjust to a given distance/focus

Control Operation "Storage of Position"

For most applications, for example process monitoring, it is convenient to store and to retrieve several camera positions in order to access predefined camera positions rapidly. A camera position is usually defined by the spatial position of the camera and the corresponding zoom and focus values. The following operations are available:

- Save actual camera position as "pos"

- Move camera to saved position "pos"
- Delete saved position "pos"

"pos" is a position identifier. Numbers or names describing each camera position seem to be the most suitable choice for the position identifiers. At the user interface these may be any kind of components, for example a list element, icons, images or moving icons.

Control Operation "Adjustment of Image Quality"

Video cameras usually have several facilities for defining the image quality. They include:

- Definition of the contrast (weak pictures are "amplified", intense contrasts are normalized)
- Colors (dull colors can be intensified, intense colors can be diminished, for example colored images can be "reduced" to black and white)
- Brilliancy (concerns the brightness of the video image in general, it is achievable by altering the share of white)
- Predominance of one color can be adjusted by the camera so that natural colors can be achieved.

The remote camera control must provide means to allow for integration of this set of commands. At this point it is important to notice that this list is different for each camera and may comprise further capabilities not included now. The camera control system allows the incorporation of further (not yet known) control commands.

Control Operation "Video Recorder"

Besides the functions for moving a remote camera the HeiCam application offers the possibility to record and playback videos captured by the camera. This can be used if a security guard or supervisor is recognizing an unauthorized access to a restricted area and the video is serving as evidence.

Video cameras with a built-in video recorder are known as "camcorders" (camera recorder). A camcorder is an integration of two different devices, the video camera and the video recorder. The HeiCam application offers control functions similar to video recorders besides its camera-oriented operations.

Typical video recorders support operations such as:

- Record/grab video images
- Freeze video (display single image)
- Reproduce video images
- Stop
- Pause

This video "processing" functions are not a specific requirement for a remote camera control. They might be part of any other video playback application.

Access Control

The acceptance of any video camera in a remote environment highly depends on the whole access control support including authorization. The system architecture should support a variety of possibilities as enumerated in this section.

With respect to the available capabilities, the access to a remote camera control can be allowed

- for all operations,
- for a subset of operations,
- for no control operations of the camera but display of video data or
- for no operation at all.

With respect to the users, machines and applications, the access to a remote camera control can be allowed

- to a person or to a group of people,
- to certain applications or
- to computers or groups of computers (i.e. for anybody working in on machines owned by a department).

If several people/applications/computers are authorized simultaneously, the access to the camera has to be synchronized. Just imagine a soccer game transmission where each spectator can control the same camera. Therefore, the possibility to control the camera has to be granted exclusively to a person/application/computer at one point of time.

User Interface

Many multimedia applications in the domain of conferencing are technology driven. Hence the user interface have not the highest priority. We see this from a different perspective, and are experienced that the users are most critical concerning this issue: The success or failure of an application like HeiCam heavily depends on the acceptance of the user interface by the user.

The user sitting in front of this computer wants to change the position of the camera in order to get another view. Therefore, he issues control commands to, for example, move, zoom, and focus the camera. He interacts with the application through the user interface of the camera control application.

The goals of the design of such a user interface are to provide

- ease of use for first-time users,
- an easy to remember mechanism for casual users,
- and a fast and effective operation technique for frequent users.

To some extent these goals are contradictory and difficult to implement. We approached this crucial issue by providing a button/menu based window interface as well as the novel "active video window" approach (as outlined in the next section).

In the novel approach the movement comprising "up", "down", "right", "left" i.e. rotations with respect to two right-angled axes of the camera are controlled by immediate interaction with the moving image [SaSt92].

Fast Response Time

Another requirement following the interaction between user and user interface of the multimedia system is the already mentioned real-time and a short response time.

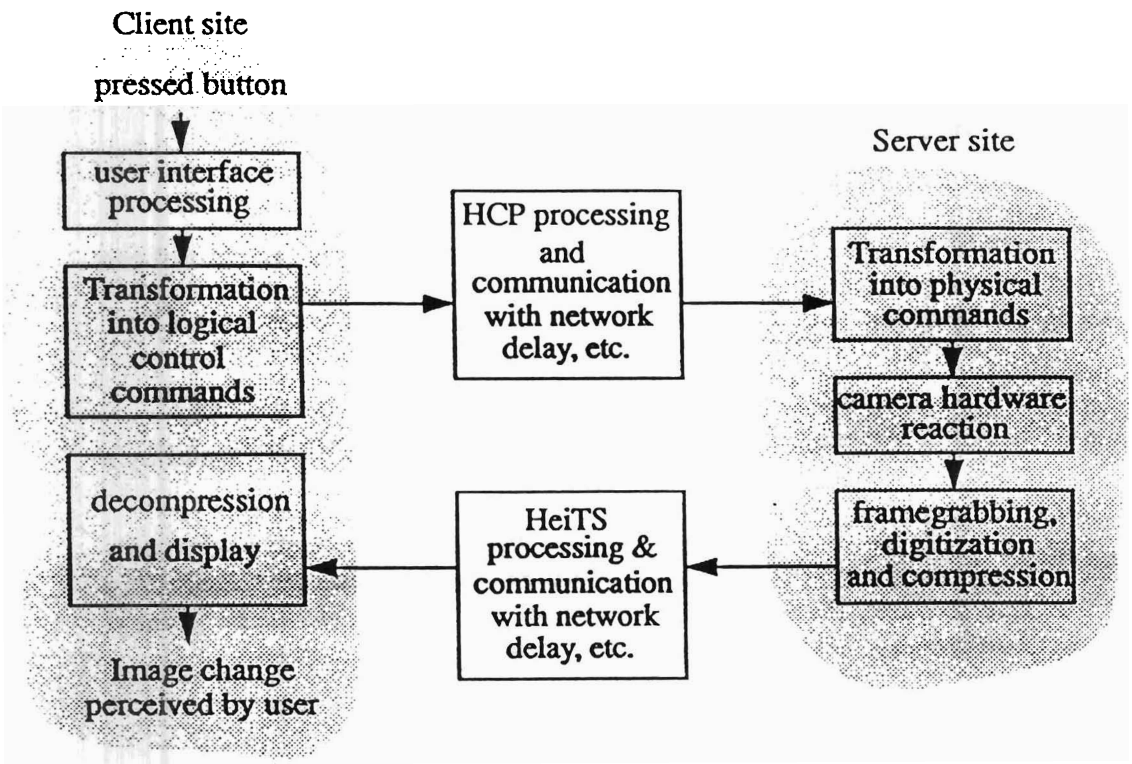

Figure 3: End-to-End Delay in HeiCam

The purpose of a remote camera control system in a multimedia environment has the possibility of controlling the "live action". It is especially the interaction between user and system that ensures a high acceptance of the overall system. This way, the user gets an immediate response of his operation shortly selected before. A multimedia system that is unable to meet these assumptions implies unwanted delays and a long response time.

These problems are described in the following short application scenario:

> A user likes to rotate his camera to the left to adjust it to a new position. Therefore, he has to select the operation "rotate camera to the left" until the wanted position is indicated on the display screen. The movement of the camera is halted by executing the stop instruction. If the moving picture is not represented in real-time but with considerable delay, the stop instruction will be executed too late. Short delays in some cases of remote camera control operations concerning the movement can be accepted. But these can not be tolerated if the camera has to be adjusted precisely, i.e. for focus operations.

HeiCam is sensible to delays as the user expects an immediate reaction after having pressed, for example a button to move the camera. The delay is defined by the whole data processing and communication chain between pressing the button and perceiving the action at the video image. This is shown in Figure 3. The user requirement study showed that a well accepted end-to-end delay in this sense should be below 500ms. Users may tolerate up to 1 second. Any value beyond 1 second turns out to disturb the daily work with HeiCam. In our recent implementation we encounter an average delay of 640ms, most of it is consumed by the used ActionMedia II hardware with the audio video kernel (AVK). The HCP of HeiCam is not very sensible to any jitter as long as commands are processed in the order they were issued by the user.

The acceptance of the remote camera control depends, in addition to the user interface handling and the provided functions, on the efficiency of the multimedia system to process time critical data streams and cope with the arising end-to-end delays.

4 Implementation Aspects

To separate the application into functional blocks, a special block of the control functions was built. The control functions are the logical part of the performance of control commands given by the user on both sites, client and server. Because of the independence of a specific camera hardware the control functions are divided into device independent control, the logical part of the control functions, and a device dependent control. Both parts are connected with a clearly defined programming interface. The part of the device dependent control is called the camera driver and performs the transmission of control codes to the camera hardware.

This existing architecture and implementation of a remote camera control fulfils all the requirements mentioned above. This realization of the requirements is described in this section as well as the used communication protocols. This section shows how HeiCam works and lightens the most critical aspects.

Camera Movements

The camera movements contain all functions for horizontal and vertical move, zoom and focus of the camera and its lens. These movements can be combined because of a similar performance of their commands. The HeiCam Control Protocol (HCP) was designed and implemented for the transmission of the control commands from client to server.

There are two possibilities to formulate a control command for a camera movement: A relative command or an absolute command. A relative command is a control command like "Move camera 180 degrees left with speed level 3". "Move camera with speed level 3 towards the position of 180 degrees" is an absolute command. The coupling of direction and speed level to one control command "Move left with speed level 3" groups logically related issues together, and it alleviates the communication paths from the transmission of at least one additional message. Using a command like this requires to give a command to stop the camera at the desired object.

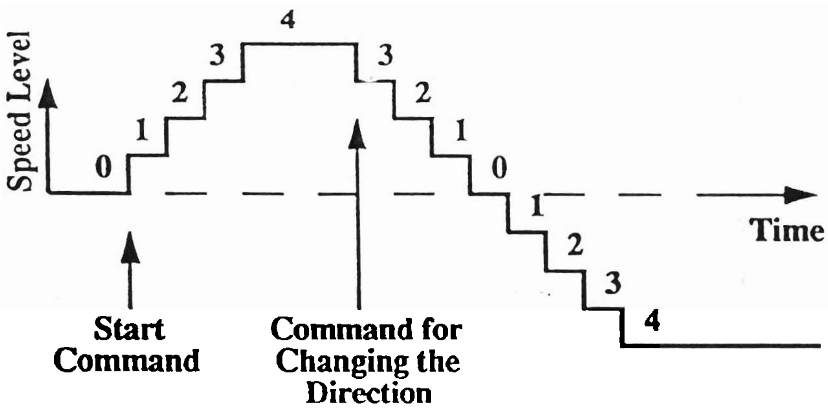

Figure 4: Acceleration and Slow Down Ramps

In general the motor of the camera hardware possess not only the state "On" or "Off", it allows the division of a movement into various speed levels. The number of speed levels depends on the used camera. The "On"/"Off" control is just a special case. The camera control engines differ in terms of the physical connection (for example electrical RS232C or infrared channel), the control codes and the semantics of these codes. Therefore, a special camera driver for each camera type is needed. Because of the inertia of the motors at the alteration of the speed level, we often experienced a certain problem: A sudden acceleration to the highest speed level in a very short period of time can cause a damage of the camera motors or demand at least a re-adjustment of the motors. This must be prevented. The introduction of ramps for acceleration and to brake down a camera movement solves this problem. This acceleration in the step by step mode avoids the damage of the motors.

Commands for changing the direction during a camera movement have the same problems. First of all, the current movement of the camera has to be slowed down, step by step. Afterwards the new direction can be taken and it can be accelerated to the new speed level. Figure 4 shows the use of ramps for acceleration and slow down of a camera movement.

Storage of Positions

The storage of positions is very specific to the used camera type. Some cameras allow to store positions by their special control hardware, others return values of their current camera positions and consequently facilitate a storage by software. Hence the handling of stored positions is also the duty of the camera driver.

The storage of a position means to save the values of the current camera position as well as values of the actual adjustment of the lens (zoom and focus). In order to address different already stored positions we introduced position identifiers. All functions which work with positions (to store, to delete, to rename or to move), demand this identifier. The fact that many users are allowed to use a distributed remote camera control and anyone shall work with his own as well as other stored positions requires meaningful identifiers. Therefore, we introduced position names instead of position numbers.

The management of these positions and the respective position names is performed by the camera driver at the server site. If a client wants to move to a desired position, then client transmits the accompanying control command and the position identifier via HCP to the server.

The server tells the camera driver to perform the movement to the named position. Depending on the camera type and the used storage technique the camera driver determines the requires values for positioning and adjustment of the lens. These values and the names of the positions are kept internally at the server in a position table.

We found it useful to keep the stored positions and their parameters in a file. They are available at any restart of the server.

Access Control

In a multiuser environment with a remote camera control as a shared resource, the access to this camera and its server has to be managed. In some scenarios it may be useful for more than one user to control the same camera concurrently. In most often encountered scenarios one user wants to exclusively control the camera, whereas many other users my be allowed to watch the video data. Therefore, the access has to be granted exclusively to one user.

The access can be allocated to

- the first client who is requesting the camera control ("first come - first serve"),
- the client requesting with the highest priority or
- the client who is next concerning some fairness principle like a long duration "time slicing".

The analysis of shared application scenarios showed that in conflict situation most/often a separate phone call between the requesting users is the most frequent way how they want to solve this resource conflict. Therefore, in the current version of HeiCam the 'first come - first serve' access control is implemented. This implementation allows the first client requesting to control the camera. The client is granted that he will not be interrupted during its time of work. This technique makes it possible for the user to work on optimal conditions for best effort but it is up to the user to release the control and to allow other users to get the access to the camera.

Authorization can be done at the server site: If the camera is released for control by the server, and the client has got access of the camera, an access-ID is allocated to the respective client. This ID must be transmitted with all control commands to authenticate the client at the server at any time. From this access-ID the server checks if a client is allowed to control the camera or not and performs or rejects the command. HeiCam is open to allow network or system management components to perform all required authentication operations.

Any unexpected termination of the controlling client application, i.e., a system crash or power failure, would mean to block for all other clients the control of the camera. The consequence is a deadlock. Therefore, a permanent check for the life of the controlling client is necessary. The client sends regular messages to the server to confirm its life. If a client has terminated and the server has not received any message from the controlling client during a certain period of time, the camera can be accessed by any other requesting client.

Video Recorder Functions

The control commands to record the current live video or to play a stored video are transmitted via HCP. The actual storage and play of the video file itself is performed by the AV subsystem. The machine where the file is recorded can be the camera client, the server or any other machine in the domain of the AV subsystem. This enables the use of file servers which are sharing repositories for stored video files captured by HeiCam and other applications. An access to these videos can be achieved by sending a respective command including servername and filename to the desired HeiCam camera server.

Camera Driver

The camera driver is the hardware-specific software module of the server program. It performs the communication between the application and the camera hardware and consequently the movement of the camera. To store a current position or to move to a stored position are camera-specific operations as well. These operations are performed by the camera driver as described above.

Each command to control the camera is represented by a corresponding function inside the driver which transfers the command to a sequence of control codes to the camera. Here we encounter a well defined programming interface to the functions of the camera driver. Hence the development of drivers for new camera types and their integration into the HeiCam application can be realized easily and fast.

The HeiCam Control Protocol (HCP)

The HeiCam Control Protocol is the protocol for the transmission of the control commands from the client to the server. Even though this protocol is not strictly time-bound the amount of transmitted data should be as little as possible. Therefore, all commands are executed at the server using ramps to accelerate and slow down.

HCP uses the Remote Procedure Call which is suited for the development of client/server applications. The RPC is able to use UDP (User Datagram Protocol) or TCP (Transmission Control Protocol) as lower layers of its own protocol. It is important to the HeiCam application that any command given by the user is executed exactly once. The reliability of TCP guarantees the fulfillment of this major requirement. The use of these standard protocols straight forward and simplified the development of the HCP.

At the transmission of a command via HCP a data structure is sent which contains all informations needed for the execution of the desired control command. This structure includes the access-ID to validate the permission to control the camera, the control command itself and the actual parameters. The server receives the transmitted data and initiates the performance of the given command depending on a valid access-ID. The result of the command, success or failure, is returned to the client which evaluates this return code.

Figure 5 shows the cooperation of client, server and the camera driver at the execution of a move command. The desired movement is "Left with speed level 3" and the current status is "Right with speed level 3".

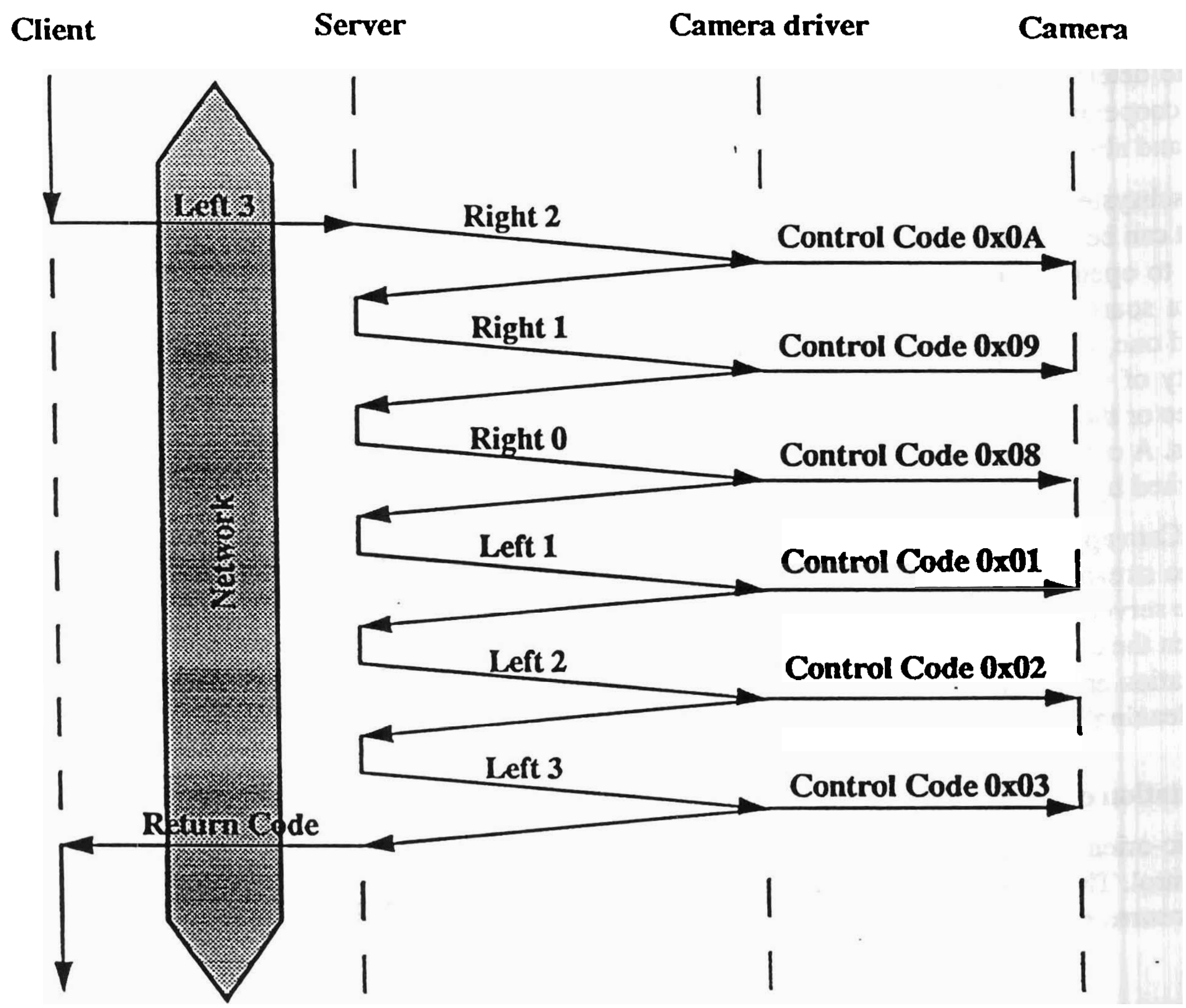

Figure 5: Execution of a Control Command

The Audio/Video Subsystem

The delivery of the audio and video data from the camera server to the connected clients is the duty of a separate audio/video subsystem. This separation of client/server program and AV subsystem allows the reuse of already developed software modules and the independence of a specific audio and video format.

The BERKOM Multimedia Collaboration Service (MMC) supports joint working in a distributed environment. It allows users to share applications and to participate in audiovisual conferences from their workstation [Alte93]. The Audio Visual Component (AVC) of the BERKOM project is interfaced by the HeiCam application. The AVC is based on HeiTS and establishes audio and video connections between servers and clients. There is one AVC on each server or client machine.

The Heidelberg Transport System (HeiTS) provides the ability to exchange streams of continuous-media data with quality of service (QoS) guarantees - where applications can specify the requirements they have for the transport service. To provide these QoS guarantees, the protocols of HeiTS are embedded into an environment which provides real-time techniques and resource management. HeiTS transfers continuous-media data streams from one origin to one or multiple targets via multicast. HeiTS nodes negotiate QoS values by exchanging flow specifications to determine the resources required - delay, jitter, throughput and reliability. Therefore, the cooperation of HeiTS and the AVC as a separate audio/video subsystem offers a powerful and already available facility for the transmission of multimedia data streams.

The AV subsystem has a clearly defined interface, the Source and Sink Control Protocol (SSCP). It can be used to establish transmissions of continuous-media data. This protocol controls how to open and to close connections by definition of endpoints. These endpoints are called data sources and data sinks, and the establishment of connections occurs between a source and one or multiple sinks. Parameters for the desired audio and video format or a certain quality of service can be attributed to any source or sink. Attributes for live or stored audio/video or the mode of the transmitted data (audio only, video only or audio and video) are parameters. A connection and transmission between source and sinks with same attributes can be established by calling the corresponding SSCP functions.

At the HeiCam application the camera server works as central control for the distribution of the audio/video streams. Each client is able to request the transmission of the video images at the server. The server opens a sink at the client site and connects this sink to the open source at the server. Then the server starts the transmission of the data stream. This management of all open communication endpoints and the established connections and transmissions allows a fast and defined "cleaning" of the AV subsystem at any shut down of the camera server.

Implementation of the User Interface

The graphic-oriented user interface of the HeiCam client is the control panel of the remote camera control. The integration of all required functions and a clear arrangement of the control elements ensures easy handling and fast interaction with the application.

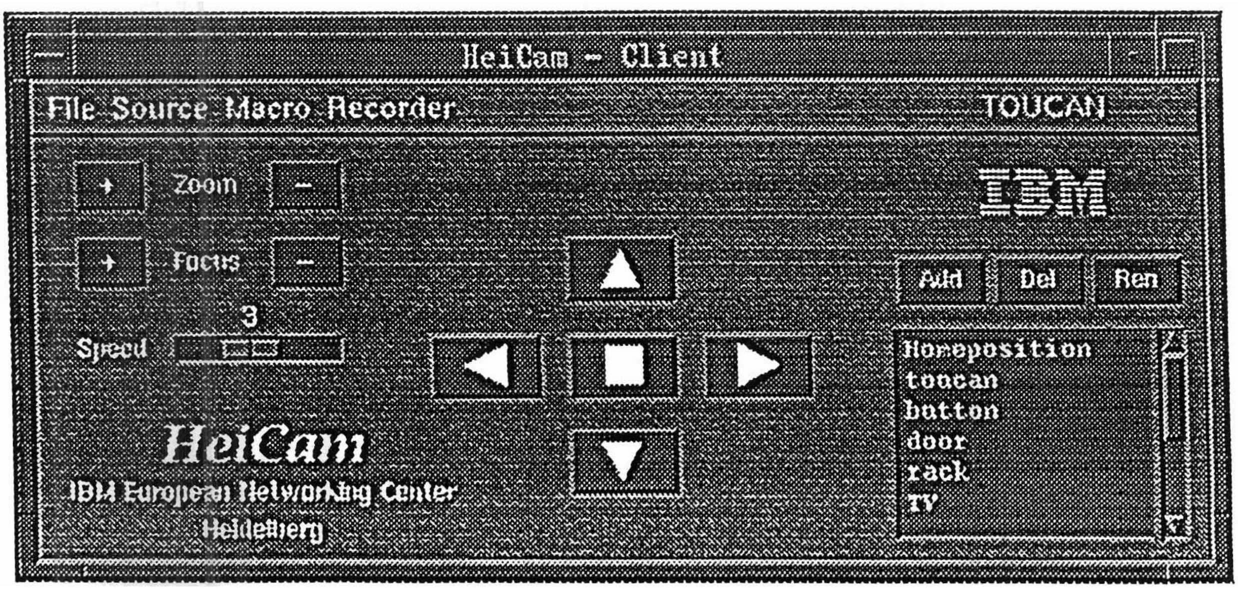

Figure 6: Traditional HeiCam-Client User Interface on AIX

Buttons for each direction of horizontal and vertical movement and for zooming and focussing the camera are integrated into the control panel. A sliderbar for fixing the desired speed level of the movements is integrated as well. A click with the mouse on a button starts the movement in concerning direction. A further click stops the started movement. The click on the stop button executes the halt of all camera movements. Using the sliderbar to change the speed level during a moving operation is possible as well. Stored positions can be added, renamed and deleted with buttons. A listbox gives a summary of all current stored positions. Using the mouse by clicking into the listbox executes the movement of the camera to the desired position.

Besides this traditional user interface our novel approach [SaSt92] means to a remote camera by clicking in the window where the video of the camera is displayed. This approach contains two variants for controlling the camera by clicking with the mouse:

- The user issues a double click at an object somewhere in the video window. Subsequently the camera is moved to a position, where this object results to be in the center of the video window.
- The user issues a single click at one side or at a corner of the video window. The nearer the click occurred at the border of the window, the higher is the speed level of the movement. I.e. a click in the upper right corner of the window causes the camera movements "left" and "up", a click nearby the center of the window causes a slowly move.

The direct interaction on the video window follows human intuition: We experienced this to be most effective and easy to use. A menu item allows the selection of the camera server. Requesting the camera control or the video image at least are the purpose of further items. To control the camera and to get the video are different actions and do not depend on each other. Menu items for starting and stopping video transmission, starting and stopping video recording and playing and the input of filenames are integrated into the user interface complete the implementation of the described user requirements.

The camera video displayed in a window is moveable and sizeable. The window can be made an icon as well.

Platform Specific Implementation

The implementation of the HeiCam application on the OS/2 platform makes extensive use of the MMPM/2, the Multimedia Presentation Manager. This extension supports the synchronization of multimedia data streams and allows the use of different devices at the same time. A programming interface, the Multimedia Control Interface (MCI), eases the developer from costly programming of the specific hardware. The management of the device drivers is done by the Multimedia Device Manager [MMPM92].

5 Experiences and Outlook

Since the start of the HeiCam project and its development in 1991 different HeiCam versions on two platforms were implemented. They were shown at many exhibitions around Europe including Cebit '92 at Hanover, Germany and the Security '93 at Utrecht, Netherlands. The field of the security and surveillance industry is changing more and more towards digital processing. Therefore, the HeiCam application finds more and more interest.

Monitoring a restricted area or surveilling a large entrance hall requires frequent changes of the camera position. The supervisor controlling the camera has to issue a number of commands to execute this changes. An automatic sequence of camera movements with stops at the desired camera positions would be easier to monitor the observed area.

A **script language** allows to record and to playback sequences of previously issued control commands. The supervisor is able to concentrate on the monitoring only. In order to record a sequence the controller has to start the recording and to issue all commands as usual. All control commands and the time between two commands is kept in a editable file. A script command to restart the programmed macro file also provides the playback of an endless sequence.

The combination of a digital remote camera control with a **set of sensors**, such as photoelectric beams or motion detectors, extends the automation of industrial monitoring systems. If an event occurred and is caught by the sensors, the camera would move directly to the scene and the video image is automatically shown at the controller's monitor and recorded to a file.

The **detection of motion** can be done by analyses of a digital video. Video compression techniques using interframe coding allow the detection of motions by an analysis of the size of these delta frames. A sophisticate analysis of a live video may also be done to control a camera by tracing a desired moving object.

HeiCam comprises the remote camera control and interfaces the transmission of live video. It includes the possibility to store and retrieve the video images in a simple way. The closely interaction with applications like HeiDi, the Heidelberg Audio and Video Distribution, offers a comfortable method to retrieve video images stored by HeiCam. Functions to play, pause rewind and fast forward the video images are part of HeiDi. HeiDi is a computer-based video player and recorder, a VCR. A sophisticated integration is reached by having a similar user interface. HeiCam and HeiDi are separate applications, but they work together as complements.

Acknowledgement

At this point we like to acknowledge Christoph Lier for his work on the OS/2 implementations and Oliver Krone for his valuable guidance of and contributions for the design and implementation on AIX.

References

[Alte93] Altenhofen, Dittrich, Hammerschmidt, Käppner, Kruschel, Kückes, Steinig: *The BERKOM Multimedia Collaboration Service*; 1st ACM International Conference on Multimedia, Anaheim, Ca., June 1993.

[Bloo92] John Bloomer: *Power Programming with RPC*; O'Reilly & Associates Inc., Sebastopol, California, February 1992.

[Ferr90] Domenico Ferrari: *Client Requirements for Real-Time Communication Services*; Technical Report TR-90-007, International Computer Science Institute, Berkeley, March 1990.

[Hehm92] Hehmann, Herrtwich, Schulz, Schütt, Steinmetz: *Implementing HeiTS - Architecture and Implementation Strategy of the Heidelberg High-Speed Transport System*; in 'Lecture Notes in Computer Science 614', Springer-Verlag, Heidelberg, Germany 1992.

[Herr92] Ralf Guido Herrtwich: *The HeiProjects: Support for Distributed Multimedia Applications*; IBM Technical Report 43.9206, IBM European Networking Center, Heidelberg, Germany, 1992.

[HeSt91] Ralf Guido Herrtwich, Ralf Steinmetz: *Towards Integrated Multimedia Systems: Why and How;* Informatik-Fachberichte, no.293: 327-342, Springer Verlag, Heidelberg, Germany, 1991.

[HeWo92] Ralf Guido Herrtwich, Lars Wolf: *A System Software Structure for Distributed Multimedia Systems*; Proceedings of the fifth ACM SIGOPS European Workshop, Le Mont Saint-Michel, France, September 1992.

[Lier93] Christoph Lier: *Design and Implementation of a Remote Camera Control for a Distributed Multimedia System*; diploma thesis at the 'Fachhochschule Wiesbaden', Germany, April 1993.

[Meye91] Thomas Meyer; *Application Scenarios for Distributed Multimedia Systems based on High-Speed Networks*; diploma thesis at the University of Mannheim, Germany, 1991.

[MMPM92] IBM Corporation: *IBM Multimedia Presentation Manager Toolkit/2, Getting Started;* June 1992.

[Niel90] Jakob Nielsen: *Hypertext and Hypermedia*; Academic Press, 1990.

[OrHa92] Robert Orfali, Dan Harkey: *Client/Server Programming with OS/2 2.0*; Van Nostrand Reinhold Company, New York, 1992.

[Sand92] Peter Sander: *Design and Implementation of a Remote Camera Control based on a Multimedia Transport System*; diploma thesis at the University of Mannheim, Germany, June 1992.

[SaSt92] Peter Sander, Ralf Steinmetz: *Method and Apparatus for Controlling a Camera*; patent, International Application, according to the Patent Cooperation Treaty PCT/EP 93/02647.

[Stei93] Ralf Steinmetz: *Multimedia-Technologie: Einführung und Grundlagen*, Springer Verlag, Heidelberg, Germany, September 1993.

[Stei94a] Ralf Steinmetz: *Data compression in multimedia computing - principles and techniques*, Multimedia Systems, vol.1 no.4: 166-172, Springer International, Heidelberg, Germany, February 1994.

[Stei94b] Ralf Steinmetz: *Data compression in multimedia computing - standards and systems*, Multimedia Systems, vol.1 no.5: 187-204, Springer International, Heidelberg, Germany, March 1994.

[StHe91] Ralf Steinmetz, Ralf Guido Herrtwich: *Integrated Distributed Multimedia-Systems*, Informatik Spektrum, vol.14, no.5: 280-282, Springer Verlag, Heidelberg, Germany, October 1991.

[Wiel93] Marcus Wieland: *Design and Implementation of a Remote Camera Control based on a Multimedia Transport System*, diploma thesis at the 'Berufsakademie Stuttgart', Germany, August 1993.

[WSS94] Marcus Wieland, Ralf Steinmetz, Peter Sander: *Remote Camera Control in a Distributed Multimedia System*; to appear in the Proceedings of the IFIP Congress '94, Hamburg, Germany, August 1994.

List of European Networking Center Technical Reports

TR 43.9214	Andreas Mauthe Werner Schulz Ralf Steinmetz	Inside the Heidelberg Multimedia Operating System Support Real-Time Processing of Continuous Media in OS/2
TR 43.9215	Emanuel Farber Thomas Schütt	The Heidelberg High Speed Transport System First Performance Results
TR 43.9301	R. Oechsle M. Graf	The Internet Protocol Family over ATM
TR 43.9302	Luca Delgrossi Frank O. Hoffmann	A Detailed Tour of ST-II for the Heidelberg Transport System
TR 43.9303	Luca Delgrossi Ralf G. Herrtwich Frank O. Hoffmann	An Implementation of ST-II for the Heidelberg Transport System
TR 43.9304	Derick Jordaan Martin Paterok Carsten Vogt	Layered Quality of Service Management in Heterogeneous Networks
TR 43.9305	L. Delgrossi; Ch. Halstrick D. Hehmann; R. Herrtwich O. Krone; J. Sandvoss; C. Vogt	Media Scaling for Audiovisual Communication with the Heidelberg Transport System
TR 43.9306	Barbara Twachtmann Ralf G. Herrtwich	Multicast in the Heidelberg Transport System
TR 43.9307	R. Steinmetz	Compression Techniques in Multimedia Systems: A Survey
TR 43.9308	Martin Bever Ulrich Schäffer Claus Schottmüller	ISO OSI FTAM and High Speed File Transfer: No Contradiction
TR 43.9309	Ralf Steinmetz	Videodatenkompression für verteilte Multimedia-Anwendungen
TR 43.9310	Ralf Steinmetz Clemens Engler	Human Perception of Media Synchronization
TR 43.9311	Luca Delgrossi; Chr. Halstrick Ralf Herrtwich; F. Hoffmann Jochen Sandvoss; B. Twachtmann	Reliability Issues in Multimedia Transport
TR 43.9312	Brian Craig McKellar Jan Roos	Buffer Management in Communication Systems

TR 43.9313	Robert Erfle Johannes Rückert Ingo Barth; Gabriel Dermler Franz Fabian; Kurt Rothermel Frank Sembach	Multimedia Document Handling - A Survey of Concepts and Methods
TR 43.9314	Martin Paterok Stefan Kaetker Carsten Vogt Luca Delgrossi, Hartmut Wittig	An SNMP MIB for the ST-II Protocol and the Heidelberg Resource Administration Technique
TR 43.9315	Luca Delgrossi Ralf Herrtwich Carsten Vogt, Lars Wolf	Reservation Protocols for Internetworks: A Comparison of ST-II and RSVP
TR 43.9316	Thomas Käppner Lars Wolf	Architecture of HeiPhone: A Testbed for Audio/Video Teleconferencing
TR 43.9317	Ralf Herrtwich Luca Delgrossi Frank Hoffmann, Sibylle Schaller	Receiver-Initiated Communication with ST-II
TR 43.9318	Michael Altenhofen Juergen Dittrich Rainer Hammerschmidt, Ralf Herrtwich, Thomas Kaeppner Carsten Kruschel, Ansgar Kueckes, Florin Spanachi Thomas Steinig, Kathrin Werner, Joerg Winckler	Implementing the BERKOM Multimedia Collaboration Service
TR 43.9319	Lars Wolf Ralf Herrtwich	The System Architecture of the Heidelberg Transport System
TR 43.9401	Jochen Sandvoss Jörg Winckler Hartmut Wittig	Network Layer Scaling: Congestion Control in Multimedia Communication with Heterogenous Netoworks
TR 43.9402	Ralf Steinmetz	Multimedia Operating Systems: Resource Reservation, Scheduling, File Systems, and Architectures
TR 43.9403	Carsten Vogt	Quality-of-Service Calculation for Multimedia Streams with Variable Bit Rate
TR 43.9404	Heinrich J. Stüttgen	Network Evolution and Multimedia Communication
TR 43.9405	Marcus Wieland Ralf Steinmetz Peter Sander	Remote Camera Control: Requirements, Concepts and Implementation
TR 43.9406	Ralf Steinmetz Hartmut Wittig	Challenges in Multimedia System Development
TR 43.9407	Lars C. Wolf Wolfgang Burke Carsten Vogt	CPU Scheduling in Multimedia Systems

TR 43.9408	Lars C. Wolf Ralf. G. Herrtwich Luca Delgrossi	Filtering Multimedia Data in Reservation-Based Internetworks
TR 43.9409	Hartmut Wittig Lars C. Wolf Carsten Vogt	CPU Utilization of Multimedia Processes: The HeiPOET Measurement Tool
TR 43.9410	Anand Gorti Martin Machler Werner Schulz	Access to SMDS over an ATM network