MULTIMEDIA COMMUNICATION, COLLABORATION AND CONFERENCING USING ALKIT CONFERO

Mathias Johanson

Alkit Communications

mathias@alkit.se

Abstract

This paper presents Alkit Confero, a software framework for real-time multimedia communication, collaboration and conferencing. The different tools comprising the framework are described and the rationale behind the design of the software architecture is discussed. Specifically, the protocols and standards that Confero adheres to are discussed. Other issues covered are multipoint collaboration support, session management and control, media encoding and processing, and error resilience for real-time video communication.

1. INTRODUCTION

Sophisticated multimedia communication systems are becoming increasingly pervasive in the workplace and in everyday life. These systems have a tremendous potential for improving communication and collaboration between individuals, both professionally and socially.

For professional use, high quality synchronous multimedia communication enables new ways of collaborative work between teams of co-workers, irrespective of geographical location. This not only reduces the need to travel, but also facilitates new ways of cooperative work, wherein the flow of information is more direct between the people involved. More efficient communication and information exchange makes it possible to increase productivity in distributed engineering and design teamwork, while improving the working conditions for the people involved. Furthermore, distributed collaborative teamwork makes it possible to use the best competence irrespective of where it is located geographically. Virtual meetings conducted over a network using sophisticated real-time multimedia communication tools give distributed teams of co-workers the opportunity to meet more often than what would be feasible with face-to-face meetings. Consequently collaborative work will be more efficient, shortening development cycles and reducing the time to market for new products and services. These benefits not only apply to engineering and design work in product development projects, but are equally viable for software development, media production, business management and various other kinds of work.

Within the health care sector, telemedicine applications make it possible for doctors to diagnose patients remotely in real time. Entirely new health care services based on teleconsultations are emerging, improving the quality of care and reducing costs.

For distance education, teleteaching solutions based on video communication, shared whiteboards and shared presentations can be realized.

These and many similar applications illustrate the need for flexible solutions to realize high quality real-time multimedia communication and collaboration. In the remainder of this document we present *Alkit Confero* – a software framework designed to meet these needs.

2. The Alkit Confero Software Framework

Alkit Confero is a software application supporting real-time multimedia communication between two or more geographically distributed participants. Alkit Confero is part of a software framework that includes a number of additional components needed to fully support high quality distributed collaborative work in distributed teams. The individual software components of this framework are described below.

Alkit Confero

The core component of the framework is the *Confero* application itself. The Confero tool supports real-time audio and video communication between two or more users. A multitude of audio and video parameters can be configured so that the tool can be used for many different applications with different requirements on media quality, and with different requirements on computer hardware, peripheral equipment and network configurations. In addition to audio and video communication, the tool also supports instant text messaging, synchronous file transfer, streaming of multimedia clips from a server, session management, and application sharing.

Alkit Invito

When a synchronous communication session is not planned in advance, a session set-up procedure is required, reminiscent of the familiar “dialing and ringing” procedure that initiates a telephone call. This function is provided by the *Alkit Invito* tool, which is a server daemon that is run in the background on workstations used for conferencing using Alkit Confero. When someone wants to initiate a communication session using Confero, a session set-up message is sent to Alkit Invito on the destination workstation. Alkit Invito will alert the callee with a ringing sound and a dialog box on the screen asking the user to either accept or decline the call. In case the call is accepted, the requested media streams (typically audio and video) will be initiated between the two workstations. Alkit Invito relies on the Session Initiation Protocol (SIP) for session initiation and control. Using SIP nomenclature, Alkit Invito is a SIP User Agent.

Alkit Servo

Alkit Servo is a multimedia streaming server capable of transmitting audio and video clips stored on disk upon request. The initiation, playback control, and termination of the streaming is controlled from a client application integrated into Alkit Confero. The media server supports QuickTime video files and RIFF (.wav) audio files. The playback can be interactively controlled from Confero using a control panel that supports pausing, playing, fast forwarding and rewinding, as well as random access to individual video frames using a positioning slider. The server also supports slow motion playback. Using Alkit Servo and Alkit Confero, a group of users can jointly view a video clip or listen to an audio clip and the playback can be interactively controlled by one of the users.

Alkit VNC

Alkit VNC is a customized version of AT&T’s Virtual Network Computing (VNC) software, allowing the participants of a collaboration session to share the view and control of any application. Originally, VNC was designed as a tool to remotely control a computer by displaying the remote computer’s desktop in a window, and conveying the mouse and keyboard control, allowing remote control of any application. In a collaboration session, it is more useful to be able to specifically select a window on the desktop, and to share the view and control of only that window. Typically, Alkit VNC is used in combination with video communication, so sharing the full desktop with VNC will mean that the video windows are also shared. Sharing video windows using VNC does not work well, since the latency in VNC is too high for real-time video, and a lot of bandwidth will be wasted unnecessarily. By sharing a single window, however, this problem is avoided. The application sharing mechanism in Alkit Confero is based on the Alkit VNC software.

Alkit Reflex RTP reflector/mixer

When a conference session including more than two participants is to be set up, some mechanism is needed to relay the multimedia data packets between the users' workstations. Simply transmitting multiple copies of the media packets is not a viable approach, since multimedia communication is bandwidth demanding, and thus multiple copies of every data packet will be an unacceptable waste of bandwidth. A more scalable approach is to use a centralized server to which all participating hosts transmit media packets that the server relays to each participating host. This reduces the number of outgoing media streams from each participant, but the load on the network that houses the reflector will of course be high. An even more scalable approach is to use multiple reflectors in a hierarchy. Multipoint communication will be discussed in more detail in section 6.

The Alkit Reflex RTP reflector/mixer is a multipoint conferencing server that performs packet relaying between the participants of a session. In the case of audio conferencing, it can also perform mixing of audio streams to reduce the bandwidth requirements further.

Besides the basic function of acting as a relay gateway for media packets, the reflector also provides functionality for simplifying NAT firewall traversal – a common problem with real-time multimedia communication. Since all media traffic passes the reflector, it is also a good point for implementing mechanisms such as access control, congestion control, session recording and playback. Moreover, the reflector provides a powerful plug-in framework allowing (possibly third party) plug-ins to be installed that can perform various processing on the media streams of multipoint communications sessions.

3. Communicating and collaborating using Alkit Confero

The components of the Alkit Confero framework described above can be used to realize a powerful collaboration and communication infrastructure for many collaborative applications. The main use of this platform is for *synchronous* collaborative work, i.e. when the information exchange and interpersonal communication happens directly, in real time, as opposed to *asynchronous* communication, such as email, which relies on a store-and-forward approach. Although the main focus is on synchronous communication, some support for asynchronous collaborative work is also provided, which will be discussed.

The communication needs of distributed collaborators can be classified into the following categories of support mechanisms:

- Direct (synchronous) interpersonal communication, including voice, video and instant messaging,
- synchronous sharing of applications and data, including multimedia objects,
- management and control mechanisms for initiating and conducting meetings, including turn-taking, content coordination, etc.

We will discuss how each of these mechanisms is supported in the Alkit Confero framework.

Synchronous multimodal interpersonal communication

In face-to-face meetings we complement our verbal communication with a multitude of visual cues, such as gestures, posture, facial expressions, etc. We also frequently use physical artifacts in the environment to support our discussions, including whiteboards, laser pointers, physical mock-ups, paper sketches and so on. When replacing face-to-face meetings with distributed meetings, we must provide equally powerful means to support the verbal communication. Using Alkit Confero, the video communication tool provides the means to convey gestures, facial expressions and the visual context of the environments. In order to be really useful, the quality of the video usually needs to be rather high, i.e. the resolution and the frame rate of the video need to be high enough to provide sufficient details and smooth dynamics. The video tool in Confero has been designed to be able to deliver very high quality video when the technical set-up allows it (i.e. sufficient network bandwidth, powerful enough computer, good enough camera). Specifically, the design approach in Confero is to be flexible so as to allow the software to be used with many kinds of hardware set-ups and network configurations, while in each situation trying to give maximum performance. Alkit Confero can thus be run on powerful workstations with high definition video cameras and high quality audio equipment, which will deliver very high quality audiovisual communication, even for multipoint sessions, providing that the network bandwidth is sufficient. Alkit Confero can also be run on laptops and netbook computers with very modest performance, using cheap built-in or USB-connected cameras and simple built-in audio devices. This will of course not give the same performance as the high end environment, but will nevertheless be useful for many applications.

To allow for this flexibility and scalability in terms of media quality, the video communication tool of Confero can be configured in many different ways to trade off between quality, network bandwidth and computational complexity to make as efficient use as possible of the available resources in every situation. A number of different video codecs (coder/decoder) with different characteristics are available, as discussed in section 4. Various quality parameters such as frame rate, resolution and compression level can be configured. Moreover, mechanisms to automatically select the best video configuration depending on the network conditions and CPU resources are available.

While not as demanding in terms of computational complexity and network bandwidth, the audio tool of Confero also supports a number of different (audio) codecs with different properties. Advanced features such as acoustic echo cancellation and silence suppressions are also available, which is useful in certain situations.

In addition to the audio and video tools, Confero also includes a synchronous text messaging tool. The main use of this tool is to send quick text messages when for some reason audio communication isn't available or desirable. In some situations, a text message is preferable to voice, for instance when conveying a telephone number or other information that needs to get across verbatim.

In multipoint conferences, when a reflector is being used for the communication, *side conversations* are supported in Confero for audio and text messages. A side conversation is a process whereby the audio or the instant messages are only delivered to a subset of the participants in a meeting. This can be useful, particularly in big meetings, when you want to make a few of the participants aware of something that is not of interest to the whole group.

The audio and video streams generated by Confero are transmitted using the RTP protocol. Various control mechanisms are implemented using the related RTCP protocol, as described further in section 4.

SESSION MANAGEMENT AND CONTROL

In addition to transmitting audio and video between a number of participants, Confero also maintains session management information that identifies each participant by a logical name and displays the status of each participating member of the conference. This information is periodically updated by means of RTCP session description (SDES) packets. The main window of the Confero GUI contains a *participants list* that identify all members of the conference session by a name that is either a person's real name (or some other descriptive text string) or an automatically generated Internet style canonical name in the form `user@host.domain`, if no descriptive name has been specified. The type of media each participant is transmitting is illustrated by an icon. The participants list is depicted in Figure 1. Note that for each member there is a checkbox for muting/unmuting the associated audio and toggling the display of video window on and off. Furthermore, for audio sources there is a volume slider for each member, making it possible to control the volume of each source independently.

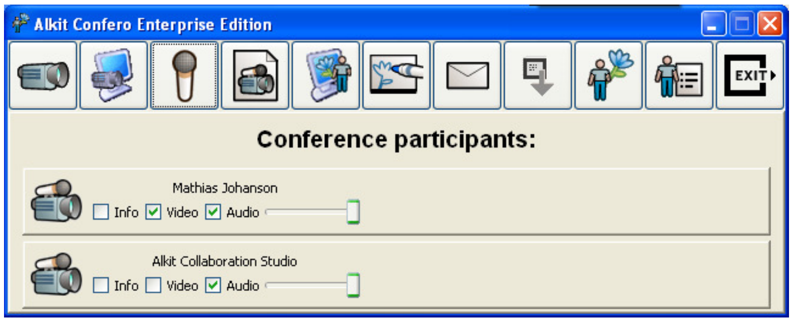

Figure 1 Session management and control interface of Alkit Confero

Session Initiation

Basically, a multimedia communication session can be initiated in one of two ways: either using explicit session set-up signaling, or without signaling. If the time and technical parameters (such as the reflector to use) for a distributed meeting has been agreed on beforehand, the participants can just start transmitting media to the other parties at the start of the meeting. For spontaneous meetings, however, an explicit signaling mechanism is needed.

Alkit Confero allows both kinds of session initiation. The explicit session set-up signaling is performed using the Session Initiation Protocol (SIP) [1]. In SIP, a user is identified by a SIP address, which to all practical purposes looks just like an email address, i.e. in the form `user@host.domain`. When a Confero user wants to initiate a spontaneous communication session with a remote user, a SIP address is specified, whereupon a SIP INVITE message is sent to the remote user's host. The invitation message will be received by the Alkit Invito SIP User Agent, running as a daemon process on the remote user's host. Alkit Invito will alert the user with a ring signal and will display a dialog box asking the user to either accept or decline the call. If the call is accepted, the exchange of RTP media data will commence. In order for this session initiation mechanism to work, the callee must have the Invito SIP User Agent running on his or her computer and the User Agent must be registered at a SIP Proxy. The Alkit Invito SIP User Agent is compatible with most commercially available (or open source) SIP Proxy Servers.

In some cases the explicit signaling procedure is not practical. For instance, in a multipoint communication session involving hundreds of potential receivers of a media stream, it would be highly impractical to have to type in hundreds of SIP addresses, send out invitation messages, and await responses from all prospective participants, before the session can start. In this situation, some mechanism external to Alkit Confero can be used to agree upon how the session should be set up , e.g. which reflector and session name to use, or which IP multicast address to use (see section 6). RTP media data is then transmitted to the reflector or multicast address, letting each potential participant independently decide whether to connect or not. To simplify this process, a session description file, based on the Session Description Protocol (SDP) [2], can be generated by Confero. This file encodes the session parameters necessary to initiate a communication session. The file can then be emailed to each person that is to be invited to the meeting, or posted on a web page used to coordinate meetings or otherwise made available to the participants of the upcoming meeting. When the file is invoked (i.e. double clicked), Confero will start and automatically connect to the meeting using the parameters of the SDP file.

Sharing Applications and Data in Collaborative Work

In a distributed collaborative work session, the need to share different kinds of data in different ways is fundamental. In Alkit Confero data can be shared among the participants of a conference session in many ways depending on the nature of the data and the usage situation.

Whiteboard

Often, the conversation in a meeting needs to be complemented with some simple drawing to illustrate an idea or visualize a concept. For such quick and simple sketching the Confero Whiteboard can be used. Much like a traditional whiteboard, the distributed digital whiteboard tool in Confero lets you draw freehand sketches and some simple geometric objects such as lines, circles and rectangles. Text can also be typed in. Various styles like color and line width can also be configured.

The design decision behind the whiteboard is that it should be very simple. It is not intended to replace your favorite computer graphics software. The design is also intended to be lightweight in terms or CPU resources needed. The whiteboard is implemented with vector graphics and requires only minimal network bandwidth.

General application sharing

When more sophisticated collaborative work needs to be performed, the general application sharing tool of Confero lets you share the view and control of more or less any desktop application. For instance, a word processor software can be shared, letting all participants of the meeting see what is being written, and if desired, take over the control of the application and start co-editing the document. Another common example of the use of the general application sharing tool is for sharing a presentation software such as PowerPoint. As yet another example, a CAD system can be shared, enabling engineers to collaboratively inspect and modify 3D designs.

This is indeed a very powerful, yet easy to use tool for supporting true real-time collaboration. Contrary to the whiteboard, the general application sharing tool usually requires a fair amount of network bandwidth and processing power from the computers, since the technology is based on taking a snapshot of the shared application window whenever it is changed, encoding the modified parts and communicating the updated pixels to the remote host. When remote control of the application is enabled, keyboard and mouse events are captured and transmitted to the remote application.

Although in principle almost any application can be shared using the general application sharing tool, some applications are not well suited to be shared this way. Applications that change the view on the screen very dynamically will typically generate very high bandwidth and consume a lot of CPU power. The prototypical example is a video player application. Another problematic kind of application to share is an audio based application (e.g. a music player), since it doesn't make much sense to only share the graphics in this case. Sharing of multimedia objects such as audio and video clips therefore requires another mechanism, which in Confero is implemented by the media streaming tool, described next.

Sharing of Multimedia Objects

To share video clips or other multimedia objects, the media streaming tool of Alkit Confero can be used to initiate collaborative playback of clips residing either on one of the participants computer, or on a dedicated multimedia server. The media clips can be audio or video files which can be instantly streamed to all participants (or a subset of the participants) of a session. The typical use for this is to jointly view a video clip or listen to an audio clip, and to be able to discuss the media content in real time.

The initiator of the media playback chooses a media clip from a file selection dialog in the Alkit Confero streaming media subtool. It is then possible to interactively control the streaming of the media from a control panel, using familiar controls such as play, pause, fast forward, rewind, slow motion playback, looping playback, and random access to individual video frames.

The streaming of multimedia data is performed by a server called Alkit Servo, which is distributed as part of the Alkit Confero installation, but it can also be run separately on a media server. The currently supported file formats of media clips are RIFF audio files (Waveform files) and QuickTime MJPEG movies. Alkit Servo transmits the media clips as RTP streams, so the streams will appear in the Confero session management interface (Figure 1) in the same way as live audio and video streams. The streaming can be either directly to a subset of the participating hosts, or through a reflector. In case of streaming through a reflector, the Alkit Reflex reflector/mixer will be automatically configured by Confero to relay the RTP packets of the media stream transmitted by Alkit Servo to all session participants.

The screen grabber

A drawback of the multimedia streaming tool described above is that the multimedia clips must be available in the correct format on a host running Alkit Servo in order to be streamed. Sometimes it is beneficial to be able to just select a video (or other) window on the screen and let the part of the screen be encoded as a video stream and transmitted like an ordinary video signal. In Alkit Confero this can be done using the screen grabber tool. The user selects the area of the screen to be converted to video by clicking and dragging out a rectangle using the mouse. The pixels of the screen in the selected rectangle are grabbed periodically (typically once every 40 milliseconds, resulting in a 25 fps video signal) and encoded using one of the video codecs available in Confero. To the remote participants, the grabbed part of the screen will appear as an incoming video stream just like any other video stream, captured from a camera or a media file.

File transfer

The conceptually simplest data sharing mechanism in Confero is the synchronous file sharing tool, which lets the users select and share any file from the hard disks of the participant's computers. The file transfer tool can actually be used both synchronously and asynchronously, depending on which is more convenient in a given situation. The typical use is that one of the participants in a meeting needs to distribute a file to all or a subset of the other participants. Instead of e-mailing the (potentially very big) file as an attachment, which is problematic for many reasons, the file transfer tool can be used to directly send the file to the other participants. When using a reflector for the communication, the file transfer is implemented by uploading the file to the reflector, which then lets the interested users download it. When not using a reflector it is implemented as a direct point-to-point TCP transfer. In the reflector case, the reflector can be instructed to leave the file on the reflector after the meeting is finished for a configurable time period, so that it is accessible also after the meeting, and possibly also available in the next meeting. In this way the file transfer tool can be said to be both synchronous and asynchronous.

4. Protocols and Standards

Standards for real-time audio and video communication over IP networks have primarily emerged from two sources: the Telecommunication Standardization Sector of the International Telecommunication Union (ITU-T) and the Internet Engineering Taskforce (IETF). Although the standards developed by these respective authorities are partly overlapping, they represent two fundamentally different standardization approaches.

ITU-T Recommendation H.323 [3] defines protocols and procedures for multimedia communication over packet networks and is an amalgam of a number of standards, including H.245 for control, H.225.0 for packetization and connection establishment, H.261 and H.263 for video coding and a number of others for supplementary services. The H.323 series standards are based on adaptations of protocols developed for the traditional circuit-switched service model of telecommunication networks (e.g. H.320 for ISDN videoconferencing and Q.931 for signaling). A significant focus is kept on interoperability and compliance.

The IETF standards framework for Internet video is a more loosely coupled set of documents, each defining a specific protocol or procedure. Furthermore, the IETF standards are more lightweight with a pronounced aim of scalability and extensibility. In contrast to the ITU-T standards, they do not define any algorithms for content coding, but include procedures and guidelines for packetization of media.

The first generation digital audio and video communication systems were designed using a connection-oriented network paradigm, based on ISDN and other circuit switched technologies (i.e. the H.320 series standards from ITU-T). The second generation systems were mainly a retrofitting of the first generation systems to a connectionless, packet-based paradigm (i.e. the H.323 framework), spurred by the proliferation of the Internet. A striking feature of the Internet is its considerable heterogeneity in terms of the equipment connected, link capacities and loss probabilities. Since basically any kind of device with a microprocessor and a network interface can be connected to the Internet, the capabilities of the terminal equipment are very different. Moreover, different access network technologies can support widely different connection speeds, and different load patterns in different parts of the network result in different congestion levels, limiting the amount of bandwidth available for an end-to-end connection. This property of the Internet stands in stark contrast to the publicly switched telephony network (PSTN), which is a much more homogeneous technological framework. Basically, all telephones support the same services and the access technology (the subscriber loop) is essentially the same everywhere. This fundamental distinction between the Internet and the PSTN is reflected in the dissimilar philosophies of the two standardization bodies involved, i.e. the IETF and the ITU-T. Whereas the ITU-T attempts to standardize terminal equipment and communication interfaces in considerable detail, with the principal aim of conformance, the IETF standards generally allow much more latitude. For video communication systems, the difference in perspective is evident when comparing the ITU-T standardization efforts (mainly the H.32x recommendations) with the corresponding IETF framework.

The Alkit Confero software framework has been designed for use in IP networks and is consequently firmly rooted in the protocols and standards developed by the IETF. Thus, Confero is free from the legacy problems associated with the first generation systems for real-time multimedia communication. However, this also means that Confero is *not* directly compatible with the large installed base of H.323 systems, although interoperation through a special gateway or bridge is possible. On the other hand, Confero is compatible and directly interoperable with other systems implementing the IETF protocols, as described below.

The decision to rely so heavily on the IETF standardization framework in the development of the Alkit Confero framework can be seen as a consequence of our perspective on real-time collaboration and conferencing as a datacom service, rather than as a traditional circuit-switched telecom service. The ongoing convergence of telephony and datacom services into a unified IP-based communication infrastructure supports this view, and the trend in IP telephony of increasingly favoring SIP to H.323 is also a strong indication in this direction. Moreover, since the goal and focus of the H.323 standard (and related ITU-T efforts) to a large extent is interoperability between end-systems from different vendors, flexibility in the design of the systems is sometimes compromised.

Communication Protocols

Fundamentally, the communication architecture of Alkit Confero and all related applications discussed in this document is based on the *Internet Protocol* (IP). The fragmentation of media into UDP datagrams is based on the *Real-time Transport Protocol* (RTP) [4]. An RTP header is inserted first in each audio and video packet, including information like timestamps, sequence numbers, source identifiers and payload type. For each payload type, a specific document known as an *RTP profile* specifies how this type of media should be fragmented into packets. The *Real-time Transport Control Protocol* (RTCP) [4] is a control protocol that allows monitoring of RTP data delivery and provides minimal session management information.

Confero uses RTP as transport protocol for audio and video. It also contains a full implementation of RTCP for session control and quality feedback. As mentioned, the RTP protocol is used on top of UDP [5], in accordance with the concept of *application level framing* (ALF) [6]. The ALF design principle is motivated by the fact that multimedia application design can be significantly simplified and overall performance enhanced if application level semantics are reflected in the transport protocol.

SIP, the *Session Initiation Protocol*, is a protocol for initiating and maintaining synchronous sessions over IP networks [1]. Using SIP signaling it is possible to invite prospective participants to a conference. SIP also includes functionality for proxy registration, forwarding and redirection services. The communication parameters, such as which media encodings are supported, which port numbers to use, etc, are encoded in a payload carried by the SIP messages using the Session Description Protocol (SDP) [2]. Confero implements the SIP and SDP protocols for session initiation and control. SIP authentication is also supported. Alkit Invito is the SIP User Agent that is used for registering at a SIP proxy server. Alkit Invito listens for incoming SIP calls and starts Confero when an invitation to a call is received. The SIP and SDP protocols are also used in Confero for setting up multipoint communication sessions on a reflector. This is described in more detail in section 6.

Media Encoding

Alkit Confero is based on an open architecture that allows many different media encodings to be used, and makes it easy to add new codecs to the application when needed. Different media encodings are appropriate in different usage situations, and depending on the network and CPU resources that are available.

Video Coding

Six different video codecs are currently implemented in Confero: MPEG-2, H.263, H.264, M-JPEG, DV and an experimental Wavelet-based codec. Each codec has its advantages and drawbacks, as discussed below.

MPEG-2

The MPEG-2 codec available in Confero is an implementation of the MPEG-2 video compression algorithm [7], developed by ISO/IEC. It is based on the discrete cosine transform (DCT), motion compensation and entropy coding. Inter-frame coding with motion compensation is performed by subdividing each frame into 16-by-16 pixels macroblocks, and then matching each macroblock to an image region of a previously encoded image, translated by a motion vector. The matching is performed over a search region in the reference frame by minimizing an error function that measures the mean amplitude difference between the two blocks. The error signal between the blocks is encoded together with the motion vector, so that the decoder can reconstruct the macroblock from the error signal, the motion vector and the prediction block. This type of video coding is known as *temporal prediction with motion compensation*, and it improves the efficiency of the coding compared to intra-frame coding that compresses each frame individually. However, the temporal dependencies introduced by the interframe coding makes the bitstream sensitive to transmission errors, since an error is propagated to the frames that are predicted from the frame where the error occurred. To reduce this problem, intra-coded frames are inserted at regular intervals in the bitstream, so that the decoder can be re-synchronized after an error. Intra-coded frames are known as I-frames in MPEG terminology, and temporally predicted frames are known as P-frames. MPEG also include bidirectionally predicted frames, B-frames, that are predicted from both previous and subsequent frames in a video sequence. B-frames introduce substantial coding latency, and are therefore not used in Confero, since the overall delays must be kept low so as not to impede interactive communication. The residual blocks resulting from the predictive coding are compressed using the discrete cosine transform, run-length coding and entropy coding. I-frames are coded without prediction using the DCT, run-length coding and entropy coding.

In Alkit Confero the MPEG-compressed video frames are fragmented into RTP/UDP datagrams as specified by the *RTP Payload Format for MPEG1/MPEG2 Video* [8].

H.263

The H.263 video codec was originally designed for low bitrate videoconferencing (i.e. below 2 Mbps) as an improvement of the H.261 codec. Similar to MPEG-2 it is a block based DCT encoding with predictive coding using (backwards) motion compensation (i.e. P-frames). The basic building blocks of H.263 can be seen as a lightweight subset of the technological components available in MPEG-2, but the bitstream syntaxes are completely different. There are more restrictions on resolutions and input formats in H.263 compared to MPEG-2. In Confero, the H.263 codec is mainly useful for low bandwidth applications when the computing platform is not powerful enough to use the H.264 codec (see below). The performance is similar to what can be achieved with the MPEG-2 codec. However, there is another benefit of having the H.263 codec implemented in Confero, namely for interoperability reasons. Lots of older videoconferencing systems (including many H.323

systems) only implement the H.261 and H.263 video codecs. Moreover, H.263 is a required video codec in the 3GPP technical specifications for IP Multimedia Subsystem (IMS) [9], so for conformance and interoperability with such systems it is of considerable interest.

H.263 video frames are fragmented into datagrams as specified by the *RTP Payload Format for ITU-T Rec. H.263 Video* [10].

H.264

The H.264 codec in Confero is an implementation of the video compression algorithm H.264/AVC, which is the result of a joint effort of the ITU-T Video Coding Experts Group (VCEG) and the ISO/IEC Motion Picture Experts Group (MPEG) [11]. The video coding standard is called H.264 within the ITU-T standards framework and MPEG-4 Part 10 using ISO/IEC terminology. The basic components of the encoder is block-based motion compensation, spatial prediction, transform coding of prediction blocks using a 4x4 integer transform, and entropy coding through context adaptive variable length coding (CAVLC) or context adaptive binary arithmetic coding (CABAC). Just like the MPEG-2 algorithm, H.264 has I-frames, P-frames and B-frames (and also SI- and SP-frame, not discussed here, used when switching between video streams), and each frame is divided into one or more slices for increased error resilience. Compared to MPEG-2, H.264 achieves on average about twice the compression performance, but at the cost of about four times as high complexity for the encoder and about twice the complexity for the decoder.

H.264-compressed video is packetized into RTP datagrams for transmission over IP, using the *RTP Payload Format for H.264 Video* [12].

M-JPEG

Another video codec available in Confero is based on the JPEG algorithm [13]. The JPEG codec encodes each frame of a video sequence independently of other frames in the sequence as a still image, in basically the same way as the I-frames in MPEG. This use of the JPEG still image compression algorithm is sometimes referred to as Motion JPEG (M-JPEG). Only spatial redundancy is reduced by the JPEG codec and not temporal redundancy. Consequently, the JPEG codec requires more bandwidth than codecs like the MPEG or H.264 codecs that reduce redundancy also in the temporal dimension. However, there are a number of advantages making JPEG suitable for high quality video communication, including low complexity, low compression delays, high image quality, and resilience to packet loss since errors will not be propagated to subsequent frames. Moreover, Confero supports storing of received video streams to disk in a QuickTime file, which can be edited and played back later. Motion JPEG is the preferential format for video editing applications, since it is possible to rearrange the images of the video sequence without having to re-compress them. Thus, the quality of the video is preserved if a video stream is saved to disk in Confero, and subsequently edited using a video editing tool like *QuickTime Pro*™ or *Adobe Premiere*™. There are several different compression schemes defined by the JPEG standard, including *lossless* JPEG and *progressive* JPEG. The algorithm used in Confero is known as the *baseline* JPEG algorithm which is based on the block-based discrete cosine transform, quantization and entropy coding.

The JPEG images are fragmented into datagrams as specified by the *RTP payload format for JPEG-compressed video* [14].

DV

Yet another video codec implemented in Confero is DV. This video coding was developed for use in consumer and semi-professional digital video cameras, and the video quality is generally considered to be very high. Similar to Motion JPEG, the DV codec uses intra coding only, and the transform coding is based on the discrete cosine transform. One of the advantages with DV is that affordable DV cameras are available, that deliver a digital, DV-encoded media stream directly to the workstation, using an IEEE 1394 serial cable, without any need for a specialized video digitizer board in the computer. In case transmission of DV-encoded video over an IP network is desired (as is possible with Confero), the use of a DV camera also relieves the computer of the computationally expensive encoding operation. On the other hand, if some other (possibly less bandwidth-demanding) encoding than DV is needed for the transmission, using a DV camera will make it necessary for the computer to transcode (i.e. to decode and then re-encode using a different codec) the video from DV to the desired encoding. This is also possible in Confero.

DV video is fragmented into datagrams as specified by the *RTP payload format for DV (IEC 61834) video* [15].

Wavelet codec

In addition to the codecs described so far, which are all based on international standards relying heavily on block-based DCT or integer transform coding, an experimental video coding algorithm based on the Wavelet transform and Zerotree quantization with built-in temporal prediction is implemented in Confero. The main benefit of this codec is that it produces a scalable bitstream that can be decoded at different quality levels in terms of resolution, frame rate and quantization distortion, depending on the available bandwidth and the capabilities of the receiver of the video stream. The codec is described in detail by Johanson [16].

Selecting which video codec to use

As we have seen, Alkit Confero supports a wide range of different video codecs, suitable in different situations. The H.264 codec can deliver high quality video at low bitrates, but at the cost of high computational complexity. The JPEG codec, on the other hand, can deliver high quality video at low computational cost, but with significantly higher bandwidth requirements. The MPEG-2 and H.263 video codecs are somewhere inbetween the JPEG and the H.264 codec both in terms of compression efficiency and complexity. The DV codec allows transmission of high quality video streams at high bandwidths, directly from affordable digital cameras, without re-compression. JPEG and DV are less sensitive to packet loss, compared to MPEG-2, H.263 and H.264, since they employ intra-coding only, and hence do not suffer from propagation errors. Finally, the experimental Wavelet codec gives a scalable bitstream that can be decoded at different resolutions depending on the processing power and display resolutions available at the receiver.

Alkit Confero gives the user considerable freedom is selecting which video codec to use in a certain situation. In some cases, such as when Confero is used in interoperability mode to communicate with some other videoconferencing tool, the video codec will be chosen during the (SIP) session set-up signaling. The capabilities of the other system will then dictate which codecs can be used, and only those will be visible for the user to select between.

While the freedom for the user to select video codec and to configure the quality parameters after preference is very powerful and desirable for advanced users, it can be difficult for inexperienced users to know which settings to use. For this reason, an automatic codec selection mechanism is available, that determines which codec is the best choice depending on the current conditions in terms of CPU load and available network bandwidth.

Video capture formats and resolutions

The resolution and frame rate of the video is configurable in Confero and depends on the type of camera and video capture device used. A wide range of cameras and capture formats are supported, ranging from simple USB cameras with modest resolution (320x240) to high definition cameras connected by high speed digital interfaces (HD-SDI). With an HD camera, the maximum resolution is 1920x1080 (usually referred to as 1080i or 1080p). The lower resolution HD format 720p (1280x720 pixels) is also supported. Traditional analog cameras (PAL or NTSC) can also be used if a suitable capture board is installed in the computer. The maximum resolutions possible with PAL and NTSC cameras are 768x576 and 640x480 respectively.

Often it is desirable to be able to switch between different resolutions that the camera in use supports, depending on different circumstances like the number of participants in a meeting or the content of the video. This is easily done on the fly in Confero without need to restart the video tool. It can also be done automatically in some cases, for instance when using the floor control mechanism described in section 7. There is also a mechanism in Confero that gives a receiver of a video stream the opportunity to change the resolution of the transmitted video. This is implemented by sending a resolution change request message from the receiver to the sender. The sender then decides whether the resolution should actually be changed or the request overridden.

When transmitting DV video over IP in Confero using a DV camera, the resolution is fixed at 720x576 for PAL DV cameras and 640x480 for NTSC DV cameras. This is because the compression is performed by the camera and Confero just packetizes and transmits the signal, and DV cameras always deliver the aforementioned resolutions. However, the frame rate is still configurable. When transcoding to a different encoding, the resolution can be reduced if desired. There are also a few other codec-imposed restrictions on which resolutions are allowed, in particular for the H.263 codec.

For some video grabber hardware, Confero supports cropping of the captured signal before transmission. This is useful for instance when a different aspect ratio of the video is desired, compared to what the camera delivers.

A number of preprocessing operations can be applied to the video signal before compression, including de-interlacing, sharpening, etc.

The frame rate of the video is configurable in Confero and can range from one to 50 frames per second (fps). Typically, around 25 fps is needed for a smooth video signal. For some cameras (typically using the 720p HD format) 50 fps can be delivered by the camera. This can be handled by Confero but such high frame rates are rarely needed.

Video decoding and rendering

At the receiving end, the video streams are decoded using the selected codec's decoding process. Since the codec can be changed on the fly in Confero, without any need for call set-up signaling and renegotiation of codec, the actual encoding used for a packet of video is determined from the RTP payload type identifier in the RTP header. After the video packets are reassembled and decoded, they are rendered to the screen. This can be done in a number of different ways depending on what the graphics board in the computer supports. The rendering can be done through Open GL, DirectShow, Windows GDI or X11 depending on the operating system platform and the capabilities of the graphics board. The size of the rendered video can be scaled if desired (e.g. by 2 or ½ horizontally and vertically or to full screen).

A number of postprocessing operations can be applied to the video after decoding (but before rendering), including mirroring, rotations, and various signal processing operations such as sharpening filters.

Audio Coding and Processing

Confero supports a number of audio codecs, with different properties in terms of audio quality and bitrate requirements. The following audio codecs are available: 16-bit linear PCM, G.722.1, DVI, GSM, G.711 (PCMA and PCMU) and Speex. The sample rate is 16 kHz for the PCM, G.722.1, DVI and Speex codecs, and 8 kHz for GSM and G.711. The bandwidth requirement is 256 kbps for the PCM codec, 32 kbps for the G.722.1 and Speex codecs, 64 kbps for the DVI and the G.711 codecs, and 13 kbps for the GSM codec. Audio is packetized into RTP datagrams following the guidelines in RFC 3551 [17]. The packetization interval is usually 20 milliseconds, i.e. each packet contains 20 milliseconds of audio samples.

In a multipoint conference session, each receiver must mix the incoming audio streams to a single stream that can be played out to the audio device. To make this possible, Confero puts the incoming audio streams in separate receiver buffers that are periodically mixed together to a playout buffer that is fed to the audio device. Since the signal levels of the audio sources can differ substantially, the amplitudes of the different audio sources can be scaled independently. Thus, the receiver of many audio sources can mix the incoming signals according to preference.

In order to avoid wasting bandwidth unnecessarily, a silence suppression algorithm is implemented in Confero which can be utilized to suppress transmission of audio when a source is silent. A threshold value can be specified in the GUI (or adjusted automatically), so that when the microphone level is below the threshold, the source is not transmitting the audio.

A common problem with audio conferencing, when not using headphones, is acoustic echo. The echoes occur when the received audio is played out through the speakers, picked up by the microphone and sent back to the originator. When the delay of the echo increases over a few milliseconds the effect is very annoying. To avoid echoes, an acoustic echo cancellation algorithm is implemented in Confero. The echo cancellation works by matching the signal picked up by the microphone to the signal recently played out, and then filtering out the difference.

Another way of preventing echoes in Confero is the *net-mutes-mic* function. When *net-mutes-mic* is activated the microphone is muted whenever an audio packet with a signal level above a certain threshold is received. This prevents audio played out through the speakers from being sent back to the originator, but also limits the interactivity of conversation.

Summary of Protocols and Standards used by Confero

Generally speaking, Confero follows the communication standards developed by the IETF wherever possible. This implies an IP-based communication platform, with the RTP transport protocol as a key component, and the SIP and SDP protocol for session set-up signaling. For video encoding and compression, Confero implements the international standards ISO/IEC 13818-2 (MPEG-2) and ISO/IEC 10918-1 (JPEG), developed by ISO[1], H.263 developed by ITU-T, H.264/AVC developed by the Join Video Team (JVT) of ITU-T and ISO/IEC, and IEC 61834 (DV) developed by IEC. The audio compression standards followed by Confero are ITU-T Recommendations G.722.1 and G.711, the ETSI standard GSM, and the IMA standard DVI. The most important standards and protocols that the tools of the Alkit Confero framework are built upon are summarized in Appendix A.

The strong focus on using open protocols and international standards whenever possible makes Confero interoperable with a multitude of other videoconferencing and VoIP tools.

5. Design and Implementation Decisions

This section discusses some of the most important design decisions that have influenced the development of the Alkit Confero software framework.

Software Architecture

The software architecture of the Alkit Confero application is based on a peer-to-peer[2] model, when considering the real-time multimedia communication functions. However, some of the subtools integrated into the graphical user interface of Confero are clearly based on a client/server architecture. In particular, the media streaming subtool consists of a client part comprising the user interface for controlling the media playback, and a server part, i.e. Alkit Servo performing the streaming operations. The application sharing support is also based on a client/server architecture, implemented using the Alkit VNC server and the Alkit VNC viewer application.

[1] Note that these ISO standards are also published by the ITU-T under the names ITU-T Recommendation H.262 and ITU-T Recommendation T.81 respectively.

[2] We use the term peer-to-peer here in its traditional meaning, as a distributed architecture where each node acts as both client and server. More recently, the term peer-to-peer (P2P) has come to denote Internet overlay architectures, such as file sharing networks.

Multiplatform Support

One of the main design decisions for the Alkit Confero components is that they should be possible to use on different hardware platforms and operating systems. Currently, the Alkit Confero application can be used on Windows PCs, on PCs running Linux, and on Apple computers running Mac OS X. The software was implemented primarily in C (some smaller parts are written in C++), using Tcl/Tk [18] for the graphical user interface. The decision to use Tcl/Tk was made to enable simple and rapid redesign of the user interface, to suit different application scenarios. Tcl, being an interpreted scripting language, is also convenient to use as a "glue" component to loosely connect many different subtools.

Loose Integration of Sub-tools

The Confero suite consists of a number of subtools, described in section 2, each with a specific focus. These tools are rather loosely coupled, making it easy to replace or omit a subset of the tools if desired. The integration of the client subtools is done using the Tcl scripting language. Furthermore, the different subtools are executed as several different processes using various interprocess communication facilities to exchange data.

Performance Issues

Real-time multimedia communication is very performance demanding. Specifically, video processing requires a lot of computational power and also puts a high load on internal data buses, for memory accesses and communication with the graphics board, etc.

To simplify multiplatform support, and to reduce cost of the overall system, dedicated hardware for signal processing has been avoided as far as possible. Thus, in order to get high performance, the signaling processing software implementations rely heavily on the host processor's capabilities in terms of specialized multimedia instructions sets, such as SIMD (Single Instruction Multiple Data) [19]. Furthermore, to improve performance on multikernel CPU architectures, Confero makes heavy use of threading, so that different execution units can perform parallel processing of tasks, for instance decoding an incoming video stream on one core while encoding video on another.

One of the main design goals for the Alkit Confero components is that the underlying technology should be as scalable as possible, both in terms of the number of participants in multipoint communication situations, and in terms of media quality scalability, depending on the network and CPU resources available. This implies that the tools should be designed in a way that makes it possible to trade off between media quality and resource consumption in a multitude of ways. Hence, for point-to-point communication in an environment where network bandwidth and CPU resources are abundant, the media quality should be able to satisfy even the highest demands. On the other hand, in situations with very many simultaneous participants, or in situations when there is a lack of bandwidth or computational resources, media quality parameters should be possible to adjust to fit the situation.

6. Multipoint Communication

In multipoint multimedia communication, something has to be done to avoid or minimize the amount of duplicate media packets transmitted over the network links interconnecting the participating hosts. If each participant were to transmit one distinct media stream to each of the other participants, it is easy to see that the network soon will be overloaded. What is needed is some form of group communication that makes it possible to transmit a data packet to a group of hosts instead of sending a unique packet destined to each participating host.

Multicast

A network mechanism implementing this functionality in IP networks is known as IP multicast [20]. A range of IP addresses (between 224.0.0.0 and 239.255.255.255) are reserved for multicast use. A multicast address is associated with a group of hosts instead of with a particular host. Group memberships are signaled specifically, using a signaling protocol called IGMP. Multicast packets are routed independently from unicast traffic using a dedicated multicast routing protocol, such as DVMRP, MOSPF or PIM.

IP multicast makes group communication much more scalable than any other approach. However, IP multicast is problematic to configure and administer, and therefore (and due to a number of other technical problems) it is not widely deployed by network operators.

Alkit Confero can use IP multicast in case the network supports it, and it is the preferred method of realizing multipoint communication sessions. However, due to the limited availability of IP multicast, there is also another possibility available, namely to use a reflector.

Reflectors

When IP multicast is unavailable, or only available in limited parts of the network, an application level approach can be taken to support multipoint communication. In this case, dedicated hosts known as *reflectors* are configured to relay the RTP media packets between the hosts participating in a conference session. Reflectors are also known as Multipoint Control Units (MCU), specifically within the H.323 framework.

For certain media streams, a synthesis of the real-time media is possible, aggregating many incoming streams into one outgoing stream. The prototypical example is audio: multiple incoming audio streams can be mixed together to one outgoing stream, saving bandwidth on the egress network connection. Albeit not as straight-forward as for audio, video streams can also be mixed. For instance, four incoming video streams can be arranged into one outgoing video stream, wherein the spatially subsampled original streams each occupy one quadrant of the output stream, typically to fit nicely on a display.

Another function that can be useful on a reflector is *media transcoding*, i.e. the conversion of a media stream from one encoding to another (or several others), typically with different compression ratios to allow for multipoint conferencing in heterogeneous network environments.

Alkit Reflex RTP reflector/mixer

The Alkit Reflex RTP reflector/mixer is a reflector that supports mixing of audio streams and also has limited transcoding functionality. When a number of hosts are to be interconnected through Reflex, each host must first signal membership of the session to the reflector, along with the preferred media types. The reflector will then configure its internal state to relay RTP packets between the hosts, using the media encodings preferred for each client host. Also, a bandwidth limit can be specified during the session set-up phase, instructing the reflector not to transmit data at a rate exceeding the limit. Reflex will then apply media transcoding to meet the bandwidth limit. For audio, the encoding will be chosen to meet the specified bandwidth limit; for video, the frame rate of each outgoing stream will be adapted to meet the bandwidth limit.

Reflex relies on the Session Initiation Protocol (SIP) for the set-up signaling between the endpoint hosts and the reflector. The Session Description Protocol (SDP) [2] is used in the payload of the SIP INVITE messages to specify the RTP payload types (i.e. the media encodings) and the bandwidth limits to be used for the session. The SDP messages can also contain an attribute instructing the reflector that a particular host is only interested in sending (or receiving) RTP data. By default, RTP data is assumed to be both sent and received by each client.

For a multipoint session through a reflector to be set up when using the Alkit Confero framework, a checkbox must be checked in the user interface of Confero indicating that the destination address specified is a reflector. A textual name identifying the session must also be specified. This name (any text string) must be agreed upon beforehand by the session participants, and is used internally in the reflector to associate incoming packets with a particular session. SIP signaling will be performed between Confero and Reflex for each participant to configure the reflector to relay RTP packets between the participants of the session. Multiple sessions can be served simultaneously by the reflector. The reflector keeps track of which incoming RTP packets belong to which session from the destination IP address, the UDP port number, and the state maintained in the reflector for each session which was established at session set-up time. The session name supplied to the reflector as part of the set-up signaling process identifies which hosts will belong to the same session. The session state is kept in the reflector until the session is terminated by the clients using SIP BYE messages.

In addition to supporting multipoint communication, Alkit Reflex also has a number of features to simplify and improve both multipoint and point-to-point communication sessions. Since the reflector is a central point that all communication passes through, it is the ideal place to perform certain functions, including access control, firewall traversal support, media transcoding, storage and playback of session, bandwidth adaptation and more. Many of these functions are implemented as plug-ins to the reflector that are dynamically loaded at run-time. The plug-in API is a public interface that can be used by third parties to develop new plug-ins, if desired.

Advanced reflector configurations

For large multipoint sessions in sparse network environments (i.e. many widely distributed participants), multiple reflectors can be used to further reduce the load on the network. This is particularly appropriate if several clusters of users are interconnected. The hosts of each user cluster then connects to their own reflector, and the reflectors are interconnected and configured to relay the packets from each host cluster inbetween them. This type of advanced reflector usage typically requires some degree of knowledge of the topology of the network interconnecting the participating hosts.

In situations where IP multicast is supported only in certain parts of an internetwork, the hosts connected to a multicast capable network segment can communicate using multicast, and then one or more reflectors can be used to interconnect the non-multicast capable hosts. Furthermore, multiple “islands” of multicast-capable networks can be interconnected using the reflector, in which case the reflector acts as an application-level multicast tunnel. The Reflex reflector/mixer supports all of the abovementioned multicast/unicast interconnection scenarios.

7. Advanced Features

Alkit Confero provides many advanced features not found in most real-time multimedia communication tools. Some of these features are described below.

Congestion Control for Video

The Internet is a connectionless best-effort network. This implies that congestion control is performed on an end-to-end basis. When network connections get overloaded, router buffers will fill up and packets are dropped. This is a signal to the transport protocol responsible for the congestion control to lower the rate of transmission. Most Internet applications use the TCP transport protocol, which uses an additive increase, multiplicative decrease (AIMD) congestion control algorithm known as *slow start with congestion avoidance* [21]. The rate of transmission is multiplicatively increased until a packet is lost. Lost packets are identified with an acknowledgement scheme. The rate is effectively halved and then increased linearly until another packet is lost. It is then halved again, and so on. Such dramatic rate changes are not problematic for ordinary data communication (like file transfers), but do not work well for continuous real-time multimedia communication, like audio and video. Moreover, TCP’s congestion control is closely coupled with the retransmission scheme, based on acknowledgements. Retransmissions are vital for reliable data communication, but for real-time audio and video, a packet arriving too late is just as bad as a lost packet, so retransmissions are generally not viable. For these reasons, TCP is not used for real-time multimedia communication applications. Such applications, including the audiovisual tools of the Confero framework, are typically based on the RTP protocol on top of the UDP protocol. Neither UDP nor RTP provides any congestion control, so this task is left entirely to the application. Consequently, most audio- and videoconferencing applications have no support for congestion control.

In Alkit Confero, four different congestion control algorithms are implemented for video streams. Audio streams are not adaptively congestion controlled, since the bandwidth of the audio is typically low enough not to cause congestion problems.

TFRC

Alkit Confero has an implementation of the TCP-friendly Rate Control algorithm (TFRC) suggested by Floyd et al. [22]. The rationale behind this algorithm is that real-time multimedia streams should be rate-controlled in a manner that makes them compete in a fair way for bandwidth with applications using TCP. Since the dramatic halving of the rate being performed by TCP is unacceptable for audio and video, a smoother variation of the rate is desired, that gives approximately the same performance as TCP over an extended time period. The TFRC algorithm relies on feedback of packet reception statistics from the receiver of a stream to the sender. In Alkit Confero, these feedback messages are implemented with RTCP Receiver Report packets. (This requires that the RTCP RR packets are transmitted much more frequently than what is usually the case.) The sender of a stream adjusts the packet transmission rate using the TCP throughput equation, stating that the throughput of a TCP connection is roughly inversely proportional to the square root of the loss rate multiplied by the round trip time [23].

DBRC

Another congestion control algorithm implemented in Confero is called the Delay-Based Rate Control algorithm (DBRC). Similar to the TFRC algorithm it relies on feedback of packet statistics from the receiver to the sender, but it uses a different equation to decide the packet transmission rate. The equation used in DBRC relies only on packet delay measurements. The motivation for using the transmission delay as a congestion indication is that prior to packets being dropped by routers in response to overloaded connections, packets will be buffered in router queues. Consequently, a network connection that is becoming congested has an increasing end-to-end delay. By responding to increases in delay, congestion can be responded to before packet loss is experienced, and hence overall loss rates can be reduced.

SMRC

The Smooth Multimedia Rate Control (SMRC) algorithm is aimed at adapting the rate of the media stream smoothly based on round-trip time measurements. This approach is similar to TFRC and DBRC, but the algorithm has a different mechanism of deciding when to increase and when to decrease the rate. This mechanism is based on calculating two thresholds, one for decreasing the sending rate and one for increasing the rate. This creates a sort of hysteresis function.

GBRC

The Goodput Based Rate Control (GBRC) algorithm works by finding out the maximum rate of the video that actually gets through to the receiver (called the goodput rate). The sender is then instructed to keep below this rate. The goodput rate is detected by gradually increasing the rate until packet loss is seen. The goodput rate is continually re-calculated in this way to respond to dynamic network conditions.

Source Quench mechanism

There is actually a fifth, very simple, congestion avoidance mechanism available in Alkit Confero. The receiver of an audio or video stream can manually choose to send a “Source Quench” message to the originator of the media stream, specifying a preferred bandwidth for the media stream in question. Upon reception of a source quench packet, the sender will adjust the sending rate to the target bitrate specified. For video streams this is done by modifying the frame rate and the quantization level of the video. For audio, the encoding is chosen depending on the requested bitrate. This type of simplistic congestion avoidance mechanism can be useful in situations where the receiver wants to control the bandwidth of the media being received, without having to instruct the sender of the media to manually reconfigure the quality and encoding parameters of the media.

The source quench mechanism in Confero requires the user to manually select the bandwidth that is desired from the sender. In Alkit Reflex, there is also a source quench mechanism implemented, that works automatically, by sending a source quench message to the sender when the packet loss rate exceeds a predefined level.

Error Correction

As previously discussed, retransmission schemes are generally not a viable approach to recover lost audio and video packets, due to the latency implications. Usually, audio and video applications have no protection against packet loss, so a lost packet will lead to the loss of an audio frame or a dropped video frame. In many situations this is not too problematic, at least not if the overall packet loss rate is low. An occasional audio click or dropped video frame is usually not perceived as too annoying. In the case of persistent loss, however, the audio and video streams can be rendered totally useless. The problem is especially pronounced for video applications, since a video frame is usually fragmented into many datagrams, and the loss of only one of those datagrams in many cases means that the whole frame will be unrecoverable, and hence the whole video frame is dropped. To make things even worse, state of the art video codecs typically rely on temporal prediction, implying that the loss of a packet in one frame can make an error propagate to several subsequent frames. In response to this, error control schemes based on forward error correction (FEC) have been proposed. FEC techniques rely on the transmission of redundant information, usually in the form of error correcting codes, from which lost data packets can be recovered.

In Alkit Confero, an adaptive FEC scheme for real-time video based on Reed-Solomon erasure codes has been implemented. The algorithm adapts the strength of the error correcting codes to the loss rate reported by the receiver.

Since the overwhelmingly dominant cause for packet loss in the Internet is congestion, simply increasing the strength of the error correcting codes in response to congestion can be contraproductive, since it increases the overall bandwidth, effectively exacerbating the congestion. Therefore, care must be exercised when using an adaptive FEC scheme.

An in-depth presentation of the adaptive forward error correction mechanism implemented in Alkit Confero is given by Johanson [24].

Chroma Keying

Chroma keying is a video technique used heavily in TV studios and movie productions. An actor (or some other object) is positioned in front of a blue (or green) screen, whereupon the blue (or green) pixels of the video are replaced by the corresponding pixels of another video signal, making it appear as if the actor (or whatever object) is part of the scene that is keyed in.

In Alkit Confero a chroma keying operation can be performed on live video transmissions. Typically, a person is placed in front of a blue screen and then a still image is keyed in as a backdrop. See Figure 2 for an example.

It is also possible in Confero to key in an incoming video stream into the outgoing stream.

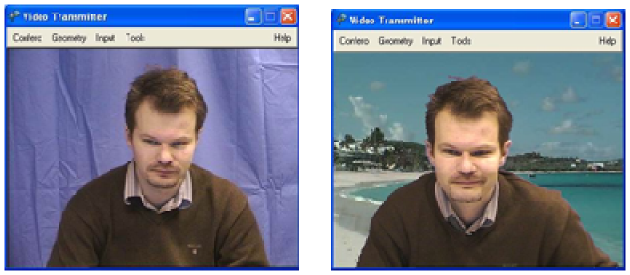
Video Transmitter
Confero Geometry Input Tools Help
Video Transmitter
Confero Geometry Input Tools Help

Figure 2 Example of chroma keying in Alkit Confero

Stereoscopic Video

In a monoscopic video communication system, the perception of true three dimensional depth is lost. By using two vertically aligned cameras with a horizontal separation of about 65 mm (modeled after the human eyes) in combination with a 3D presentation system, the perception of depth can be preserved.

In Alkit Confero an outgoing video stream can be tagged as either the left or right viewpoint of a stereo pair. A stereoscopic rendering mechanism is also implemented, based on time-division multiplexed presentation of the left- and right images, synchronized with a pair of active liquid crystal shutter glasses. Passive polarized eyewear in combination with polarized projectors can also be used. Autostereoscopic displays that don't require any eyewear are also emerging. The result is that the two video signals transmitted over the network are merged into one stereoscopic scene at the remote end.

Stereoscopic video can be useful in a multitude of applications, including remote control of robots, remotely guided surgery, and high-end telepresence systems. The design of the stereoscopic video communication mechanism implemented in Alkit Confero is described in detail by Johanson [25].

Spatialized audio

In multipoint communication sessions, with multiple incoming video and audio streams, Confero supports directional audio playback by individually panning the incoming audio streams between the left and right speaker. Moreover, the panning can be done automatically from the position of the corresponding video window on the screen, which gives the user the impression that the voice of a participant is coming from the position on the screen where the video of the person's face is rendered.

Graphical Annotations of Video

In some situations it might be useful to be able to overlay graphical annotations on the live video stream being transmitted. For instance, in a distributed engineering scenario, where two or more engineers or designers collaborate over a network showing each other physical mock-ups of early prototypes by placing them in front of the video camera, it might be highly useful to be able to draw an annotation, like a pointer or a circle, to highlight a certain part of the video signal and support the discussion. Alkit Confero supports graphical annotations of live video signals. Two types of annotations are available: circles and pointer arrows. Any number of annotations can be drawn onto a video signal, and they can be individually moved around and erased.

Firewall Traversal Support

Generally, firewalls do not let UDP packet streams, such as audio and video streams, originated from outside the firewall to pass through. This complicates the use of tools such as Alkit Confero, since the firewalls must be manually configured to let the traffic pass. In firewalls employing network address translation (NAT), port redirection must be configured on the firewall for each user that wishes to use audio and video conferencing.

To aid the users in this situation, Alkit Confero provides an automatic firewall traversal function that in many cases makes firewall traversal transparent to the user. However, a requirement is that one of the participants of a point-to-point communication session is not located behind a firewall, or that the reflector being used for a multipoint (or point-to-point) communication session is not located behind a firewall. If one of these conditions is met, enabling firewall traversal support on the hosts located behind firewalls will make the traffic pass through the firewalls without manual configuration.

The technique Alkit Confero uses to accomplish this is to let the host that is behind a firewall transmit an occasional "dummy" packet with UDP source port number equal to the port number to which Confero expects to receive audio or video RTP/UDP packets. The destination port is a well-known port that Confero listens to to detect firewall traversal support packets (i.e. the "dummy" packets). These packets will pass through the firewall (since they are originated from inside the domain the firewall protects) and will arrive at the host or reflector not located behind a firewall. When such a packet is received, the source UDP port number and the source IP address are registered, whereupon the outbound audio and/or video streams subsequently are transmitted with the destination address and port equal to the observed IP address and UDP port. This will be the port and address that the firewall assigned this packet stream in the network address translation process. Packets destined for this address/port are considered responses to the "dummy" packets transmitted, and are let through the firewall, with the destination address and port translated to the IP address of the host running Confero and the port number used by Confero for video or audio. Thus, the video and audio packets have automatically traversed the firewall.

Since the state in the firewall is refreshed after some time, the "dummy" packets are retransmitted periodically.

Remote Camera Control

A remote camera control interface is integrated into Alkit Confero. A camera supporting the VISCA standard for camera control over a serial line (e.g. the Sony EVID-31 camera or the Sony VID-P150 document camera) can be remotely controlled by someone receiving the transmitted video stream. A background daemon called *rcamd* is used to communicate pan, tilt and zoom commands to the camera. The remote Confero instance is connected to the *rcamd* daemon whenever the user enables the remote camera control. The camera is interactively controlled by clicking the left mouse button while the mouse pointer is in the video window corresponding to the camera that is being remotely controlled. Holding the mouse button down and dragging the mouse pointer will move the camera in the desired direction. The speed of the movement is controlled by the distance the mouse pointer is dragged from its original position. Zooming in and out is performed much in the same way using the middle mouse button instead of the left.

Floor Control

The floor control mechanism in Confero is designed for large multipoint conferences (i.e. more than 10 participants), when there is a need to control who is the current speaker, and to configure the video settings and screen layouts accordingly. The mechanism works by letting a user designate who is the current speaker. A number of events are then triggered, such as setting the appointed speaker's video resolution and bandwidth limit to predefined values. The typical situation is that the current speaker's video is set to high resolution and high bandwidth, whereas the other participants' video streams are set to a lower resolution and lower bandwidth limits. The layout of the video windows on the screen can also be automatically changed when the floor is given, so that the current speaker for instance is always at the top left of the screen.

Motion and Contrast Detection

For some video applications it is useful to be able to trigger various events when there is a change in motion or contrast in a video signal. For instance, a surveillance application could start transmitting video only when there is motion and stop transmitting when the video image is static. For such applications, the motion and contrast detection features in Confero let the user configure a number of actions to be triggered when there is a change in the motion or contrast levels in an “activity zone,” defined as a part of the video frame or the whole video frame. Actions can be triggered both locally (at the video sender’s side) and remotely (at the receiver’s side). Typical local actions are to toggle transmission on and off, to change the frame rate, or to toggle a privacy filer on and off. Examples of remote actions are to hide or show the video window or to scale the size of the video window. There is also the opportunity of designing user-defined actions to be undertaken. This is done by supplying a TCL script which is executed by Confero in response to a motion or contrast event.

8. Summary

This document has given an overview of the Alkit Confero software framework for real-time multimedia communication, collaboration and conferencing. The technological components comprising the framework were described, and the most important design decisions in the development of the software tools were discussed. Specifically, the network and transport protocols, the media encodings, and the protocols and mechanisms for session management and control were discussed. Moreover, the possibilities and challenges of realizing multipoint conferencing sessions, either using IP multicast, reflectors, or a combination of both, were explored. Some of Confero's more advanced features, including congestion control, error correction, stereoscopic video, spatialized audio, floor control and firewall traversal support were also discussed.

References

[1] J. Rosenberg et al., “SIP: Session initiation protocol,” IETF RFC 3261, June 2002.

[2] M. Handley, V. Jacobson and C. Perkins, “SDP: Session description protocol,” IETF RFC 4566, July 2006.

[3] ITU-T Recommendation H.323, “Packet based multimedia communication systems,” International Telecommunication Union, Telecommunication Standardization Sector, Geneva, Switzerland, February 1998.

[4] H. Schulzrinne, S. Casner, R. Frederick and V. Jacobson, “RTP: A transport protocol for real-time applications,” IETF RFC 3550, July 2003.

[5] J. Postel, “User datagram protocol,” IETF RFC 768, August 1980.

[6] D. Clark and D. Tennenhouse, “Architectural considerations for a new generation of protocols,” Proceedings of ACM SIGCOMM'90, pp. 200-208, September 1990.

[7] ISO/IEC International Standard IS 13818-2, MPEG-2 Video, 1996.

[8] D. Hoffman, G. Fernando, V. Goyal and M. Civanlar, “RTP payload format for MPEG1/MPEG2 video,” IETF RFC 2250, January 1998.

[9] G. Camarillo and M. García-Martín, “The 3G IP Multimedia Subsystem (IMS): Merging the Internet and the cellular worlds,” John Wiley & Sons, 2006.

[10] J. Ott, C. Bormann, G. Sullivan S. Wenger and R. Even, “RTP payload format for ITU-T Rec. H.263 Video,” IETF RFC 4629.

[11] ITU-T Recommendation H.264, “Advanced video coding for generic audiovisual services,” March 2009.

[12] S. Wenger, M.M. Hannuksela, T. Stockhammer, M. Westerlund and D. Singer, “RTP payload format for H.264 Video,” IETF Internet draft, IETF RFC 3984.

[13] ITU-T Recommendation T.81, “Information technology – Digital compression and coding of continuous-tone still images – Requirements and guidelines,” September 1992.

[14] L. Berc et al., “RTP payload for JPEG-compressed video,” RFC 2435, October 1996.

[15] K. Kobayashi et al., “RTP payload format for DV (IEC 61834) video,” RFC 3189, January 2002.

[16] M. Johanson, “A scalable video compression algorithm for real-time Internet applications,” Proceedings of the Fourth EURASIP Conference on Video / Image Processing and Multimedia Communications, Zagreb, Croatia, July 2003.

[17] H. Schulzrinne and S. Casner, “RTP profile for audio and video conferences with minimal control,” RFC 3551, July 2003.

[18] J. Ousterhout, “Tcl and the Tk toolkit,” Addison-Wesley, 1994.

[19] M. S. Tomassi and R. D. Jackson, “An evolving SIMD architecture approach for a changing image processing environment,” DSP & Multimedia Technology, pp. 1-7, October 1994.

[20] S. Deering, “Multicast routing in a datagram internetwork,” PhD Thesis, Stanford University, December 1991.

[21] V. Jacobson and M. J. Karels, “Congestion avoidance and control,” Proceedings of the Sigcomm '88 Symposium, vol. 18(4): pp.314–329, Stanford, CA, August, 1988.

[22] S. Floyd, J. Padhye and J. Widmer, “Equation-based congestion control for unicast applications,” Proceedings of ACM SIGCOMM'00, May 2000.

[23] J. Padhye, V. Firoiu, D. Towsley and J. Kurose, “Modeling TCP throughput: A simple model and its empirical validation,” SIGCOMM Symposium on Communications Architectures and Protocols, August 1998.

[24] M. Johanson, “Adaptive forward error correction for real-time Internet video,” Proceedings of the 13th Packet Video Workshop, Nantes, France, April 2003.

[25] M. Johanson, “Stereoscopic video transmission over the Internet,” Proceedings of the Second IEEE Workshop on Internet Applications (WIAPP'01), San Jose, CA, July 2001.

APPENDIX A - PROTOCOLS AND STANDARDS

Protocol/standard	Formal name	Standardization body	Purpose	Tools
Internet Protocol (IP)	RFC 791	IETF	Network protocol	All
User Datagram Protocol (UDP)	RFC 768	IETF	Datagram service, used for multiplexing of RTP media streams, and more.	Confero video and audio tools, Servo media streaming, Alkit Invito, Reflex
Transmission Control Protocol (TCP)	RFC 793	IETF	Transport protocol for services requiring reliable transport.	Confero text messaging, Confero file transfer, Alkit VNC, Reflex
Real-time Transport Protocol (RTP)	RFC 3550	IETF	Transport protocol for realtime data (audio, video)	Confero video and audio tool, Servo media streaming, Reflex
Session Initiation Protocol (SIP)	RFC 3261	IETF	Session initiation	Alkit Invito, Confero SIP client, Reflex
RTP profile for audio and video conferences with minimal control	RFC 3551	IETF	Media packetization rules	Confero audio and video tools
Session Description Protocol (SDP)	RFC 2327	IETF	Session description	Alkit Invito, Alkit Confero, Reflex
RTP payload format for MPEG1/MPEG2 video	RFC 2250	IETF	Media packetization	Confero video tool
RTP Payload Format for ITU-T Rec. H.263 Video	RFC 4629	IETF	Media packetization	Confero video tool
RTP payload format for H.264 Video	RFC 3984	IETF	Media packetization	Confero video tool
RTP payload for JPEG-compressed video	RFC 2435	IETF	Media packetization	Confero video tool, Servo
RTP payload format for DV (IEC 61834) video	RFC 3189	IETF	Media packetization	Confero video tool
MPEG-2	ISO/IEC 13818-2	ISO/IEC	Video compression	Confero video tool
H.263	ITU-T Rec. H.263	ITU-T	Video compression	Confero video tool
H.264/AVC	ITU-T Rec. H.264 / ISO/IEC 14496-10 AVC	JVT (ITU-T & ISO/IEC)	Video compression	Confero video tool
JPEG	ISO/IEC 10918-1	ISO/IEC	Image/video compression	Confero video tool
DV	IEC 61834	IEC	Video compression	Confero video tool
G.711	ITU-T Rec. G.711	ITU-T	Audio compression	Confero audio tool
G.722.1	ITU-T Rec. G.722.1	ITU-T	Audio compression	Confero audio tool
GSM 06.10	ETS 300 961	ETSI	Audio compression	Confero audio tool
DVI	ADPCM DVI	IMA	Audio compression	Confero audio tool

Protocols and standards followed by the Alkit Confero framework