Transparent Computer Shared Cooperative Workspace (T-CSCW) Functional Specification

John C. Checco

Abstract:
The purpose of this paper is to define the functional specifications for creating the Transparent CSCW model as a consumer based product.

Rev #	Date	Description
0.10	01-10-95	Initial Draft
0.20	01-25-95	Research References
0.30	02-01-95	Multi-Point Architectures
0.40	03-03-95	Details about Connectivity
0.50	04-10-95	Prototype Overview, Logic Flow, Acceptance Requirements, and I/O Control Commands
1.01	05-30-95	Proposed Possible Camera Video as Grayscale
2.01	04-20-96	Replaced dependency upon ISDN with suggestions for other technologies (HDSL).
3.00	08-13-96	Split document into Overview, Architecure, Functional, and Prototype specifications.

Table of Contents

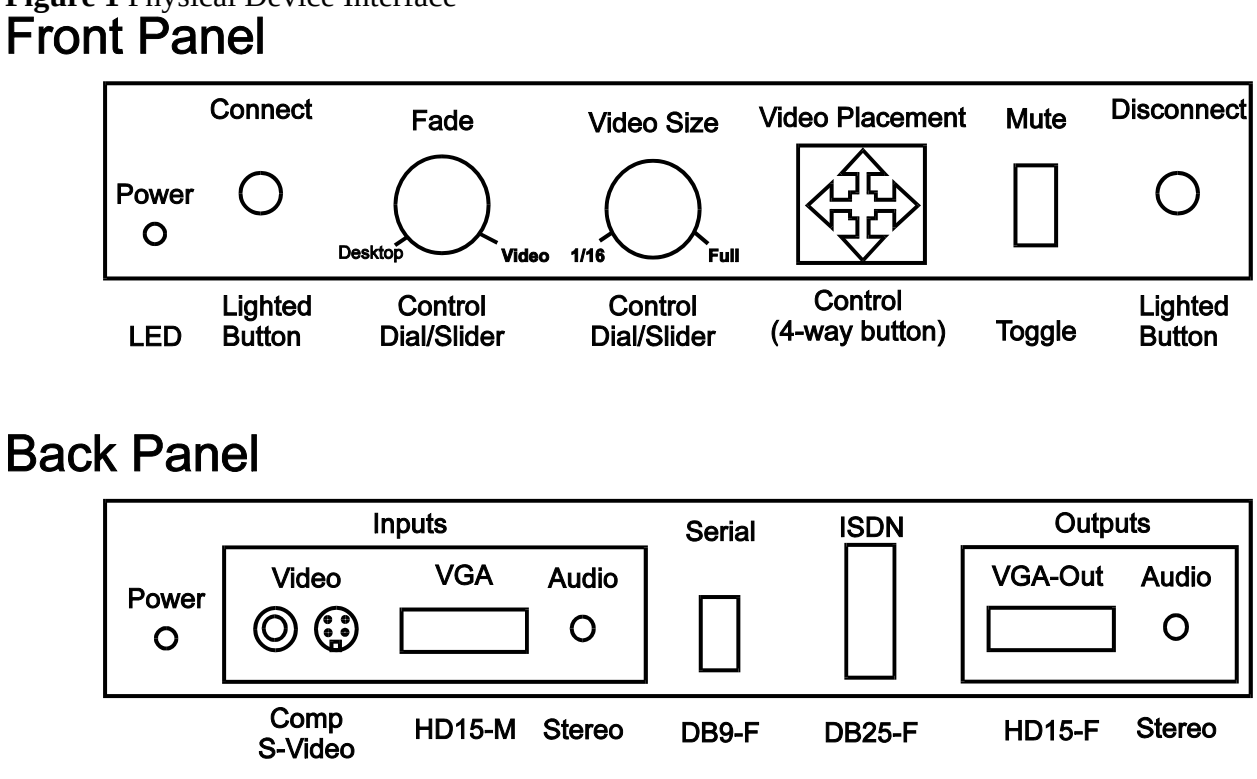
Transparent Computer Shared Cooperative Workspace (T-CSCW) Functional Specification

John C. Checco
John_Checco@compuserve.com

1 Hardware Overview

For a custom hardware solution, each workstation would consist of a single hardware peripheral that takes an SVGA input, an NTSC input, an audio input, an SVGA video output, a serial RS232C I/O line and an I/O line to be connected to an HDSL device. This box should make it easy for any user to hook up (refer to **Figure 1**).

Figure 1 Physical Device Interface

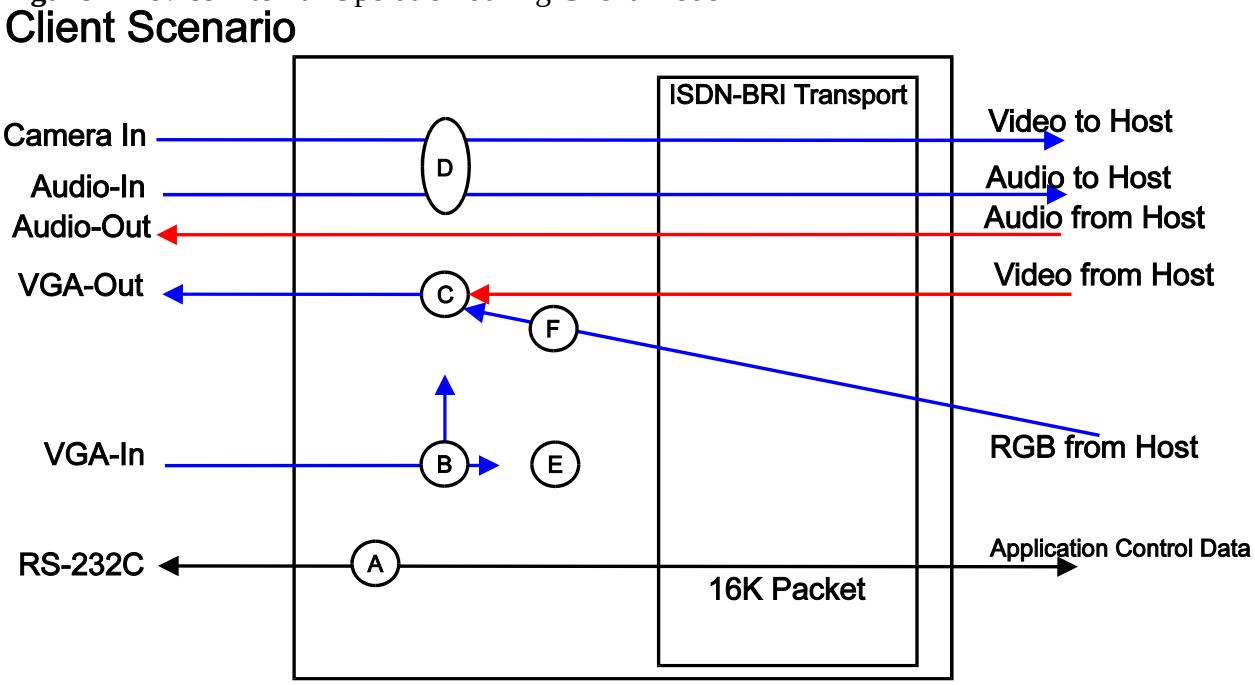


The front panel controls should allow for manual image fading (from SVGA to Video Image), manual interpolation of Video image from 1/16 rectangle to a full size overlay of the incoming

SVGA resolution, manual control over placement of Video image within the SVGA screen. Also, the box should be equipped with a video/audio mute button.

The single RS232C serial connection would receive commands from the software driver that would control the communications connection, mixer inputs (for switching between host and client modes), VGA passthru (for switching between normal operation and shared operation), and inter-computer communications (e.g. passing of keyboard and mouse commands). In addition, I/O commands should exists to allow the software to emulate any of the front panel controls (i.e. video size, position, fade level, and mute). More detailed flows of the CSCW device is shown in the following scenarios (refer to **Figure 2** and **Figure 3**):

Figure 2 Device Internal Operation during Client Mode

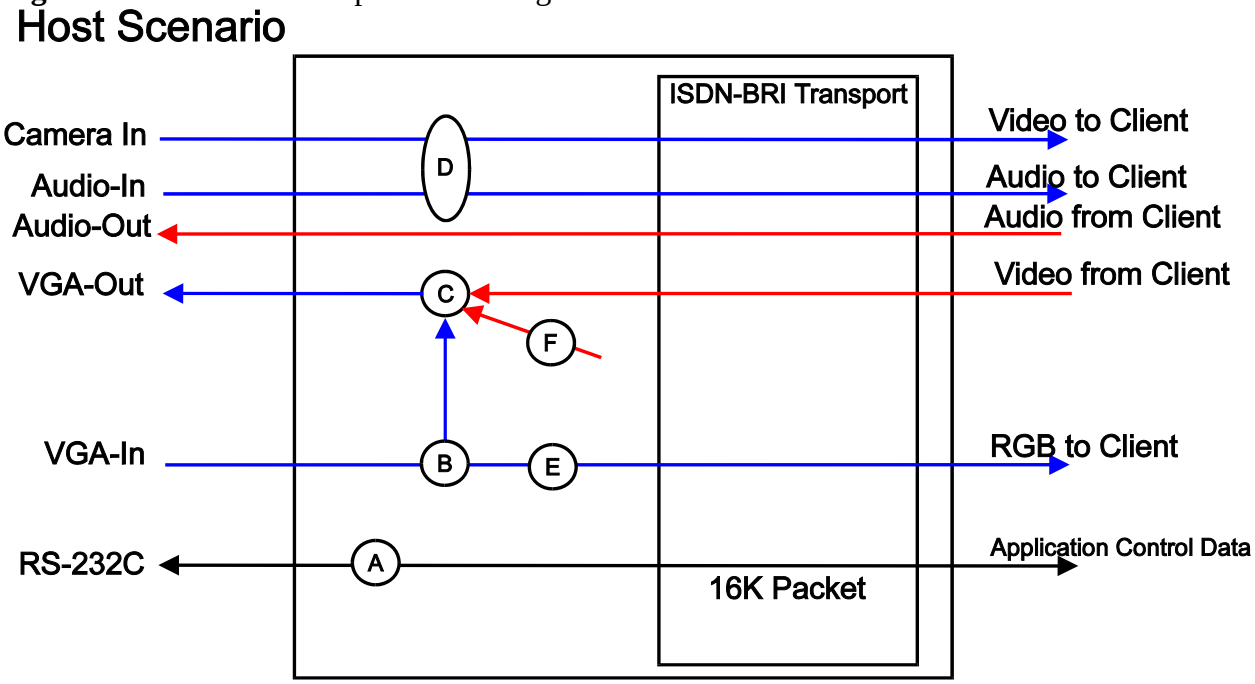


A: Serial input accepts various predefined messages for:
1 - passing application-specific data through ISDN transport between computer serial ports at both ends.
2 - controlling connection parameters of ISDN transports
3 - controlling inputs to mixer (section C)

B: VGA-In is looped to VGA-out under following conditions:
1 - no call is active, or
2 - call is in desktop sharing mode and this desktop is being shared (which then defines this station as host)

C: Video from Host is mixed to VGA-out if call is active.
D: Camera and Audio are sent to Host if call is active and mute is off.
E: VGA-In is sent to Host if call is in desktop sharing mode and this desktop is being shared.
F: RGB from Host is mixed to VGA-out if call is in desktop sharing mode and Host desktop is being shared.

Figure 3 Device Internal Operation during Host Mode



A: Serial input accepts various predefined messages for:
1 - passing application-specific data through ISDN transport between computer serial ports at both ends.
2 - controlling connection parameters of ISDN transports
3 - controlling inputs to mixer (section C)

B: VGA-In is looped to VGA-out under following conditions:
1 - no call is active, or
2 - call is in desktop sharing mode and this desktop is host

C: Video from Client is mixed to VGA-out if call is active.
D: Camera and Audio are sent to Client if call is active and mute is off.
E: VGA-In is sent to Client if call is in desktop sharing mode and this desktop is being shared.
F: RGB from Client is mixed to VGA-out if call is in desktop sharing mode and Client desktop is being shared.

2 Hardware Logic Flow:

The following rules are designed to allow the CSCW device to communicate with a user's computer. In the following scenarios, "BOX" refers to this CSCW device. All data sent to "local computer" is through the RS232C serial port. All data sent to "remote computer" is sent via the communications connection. At this time, the boundary which separates intelligent hardware control and software control is negotiable. The current logic flow diagrams have been designed to allow the CSCW device to act as a stand-alone video-phone connection, if no computer intervention is desired.

2.1 Default mode:

The BOX, by default, will ignore any incoming audio/video signals (local and remote), and will have the default video mix set to allow the local SVGA signal to pass through to SVGA output unchanged. The BOX, when powered on or reset, should always place the connected communications device to auto-answer mode.

2.2 Call Initialization

The BOX may initiate calls for software driven connections or manually operated connections.

2.2.1 Manual Operation

If the computer is not powered on, or if there are no software drivers loaded, the user must press the "Connect" button on the BOX to notify it that a manual connection is to be made. While user is manually connecting (via dial-up) the remote destination, the BOX waits for a communications connection to be detected.

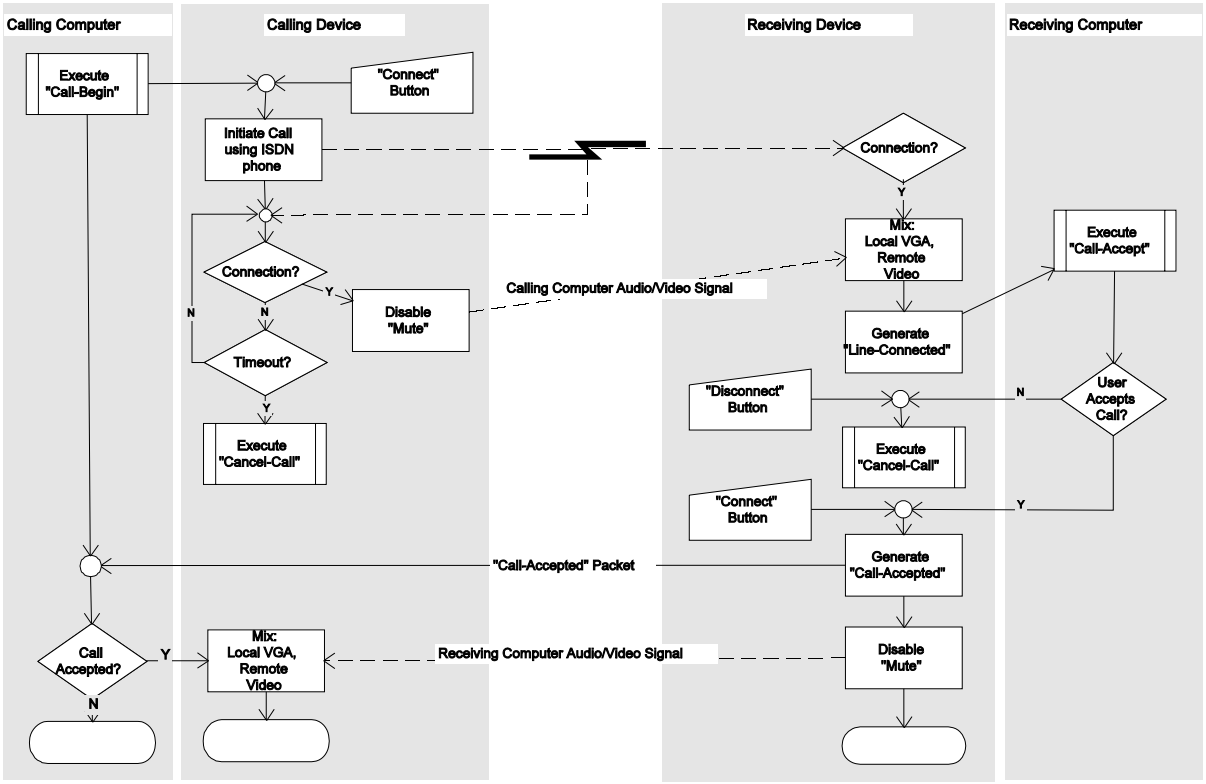
2.2.2 Software Operation

If software drivers are loaded, a user may initiate a call via the software. The software then sends an IOCTL command representing "Connect" along with the connection parameters to the BOX. (These connection parameters are dependent on the transport method used by the BOX.) The BOX then commands the communications device to dial-up and connect to the remote destination. The BOX then waits for a communications connection to be detected.

Once a connection is established, the local BOX sends a handshake signal to the remote connection. Upon receipt of an acknowledgement signal, the BOX will start encoding and sending audio/video signals to the remote connection. In addition, a "Call-Accepted" is sent to the local computer (in the case that the software drivers are running and awaiting a dialog box action). If no connection is made within a specified time (timeout threshold), the call is canceled and a "Call-Timeout" command is sent to the local computer.

At this point, the call has been completed on the caller's end, as shown in **Figure 4**.

Figure 4 Call-Initialization Flow



2.3 Communications Connection Detected:

If the BOX is not initializing a call, i.e. default mode, the BOX should be passively detecting all connections made through the communications device. Once a connection is established, the BOX will await a handshake signal identifying the remote connection as a video/audio connection. This handshake code can either be the "CSCW-Handshake" synchronization command (described later) or some standard H.320 handshake code. Upon receipt of either handshake, the BOX returns the appropriate acknowledgement signal.

Again, if this BOX did not initiate the call, it will start accepting and decoding remote audio/video signals, mix the local SVGA and remote video signals to the SVGA output, allow remote audio to be output, and an IOCTL command representing "Line-Connected" is

sent to the local computer. The user, at this time, should be able to see and hear the caller through their computer SVGA monitor. However, the caller cannot see nor hear the user.

2.3.1 Manual Operation

If the computer is not powered on, or if there are no software drivers loaded, the user can only accept or refuse the call by manually pressing the "Connect" or "Disconnect" buttons on the BOX, respectively.

Upon receiving a "Connect" button press, an IOCTL command representing "Call-Accepted" and audio/video signals are encoded and sent to the caller. In addition, a "Call-Accepted" is sent to the local computer (in the case that the software drivers are running and awaiting a dialog box action.)

Upon receiving a "Disconnect" button press, the call is actually disconnected. In addition, a "Call-Refused" is sent to the local computer (in the case that the software drivers are running and awaiting a dialog box action.)

2.3.2 Software Operation

If the software drivers are loaded, a dialog box is shown asking the user to accept or refuse the call. The user may then accept or refuse this call by selecting the "Accept" or "Refuse" options in the dialog box, respectively. (The user still has the option to accept or refuse the call by manually pressing the "Connect" or "Disconnect" buttons on the BOX, respectively.)

Upon the user "accepting" the call via software, an IOCTL command representing "Connect" is sent to the BOX. The BOX, at this point should follow the actions associated with manually pressing the "Connect" button (described above in "Manual Operation").

Upon the user "refusing" the call via software, an IOCTL command representing "Disconnect" is sent to the BOX. The BOX, at this point should follow the actions associated with manually pressing the "Disconnect" button (described above in "Manual Operation").

At this point, the call has been completed on the receiver's end. An overview of this logic is found below in **Figure 4**.

2.4 Call Termination

The BOX may terminate a connection via a software command or manual operation.

2.4.1 Manual Operation

If the computer is not powered on, or if there are no software drivers loaded, the user may press the "Disconnect" button on the BOX. This will cause the BOX to command the communications device to disconnect.

2.4.2 Software Operation

If software drivers are loaded, a user may initiate a call via the software. The software then sends an IOCTL command representing "Call-End". This command directs the BOX to act as if the "Disconnect" button has been pressed in the above section.

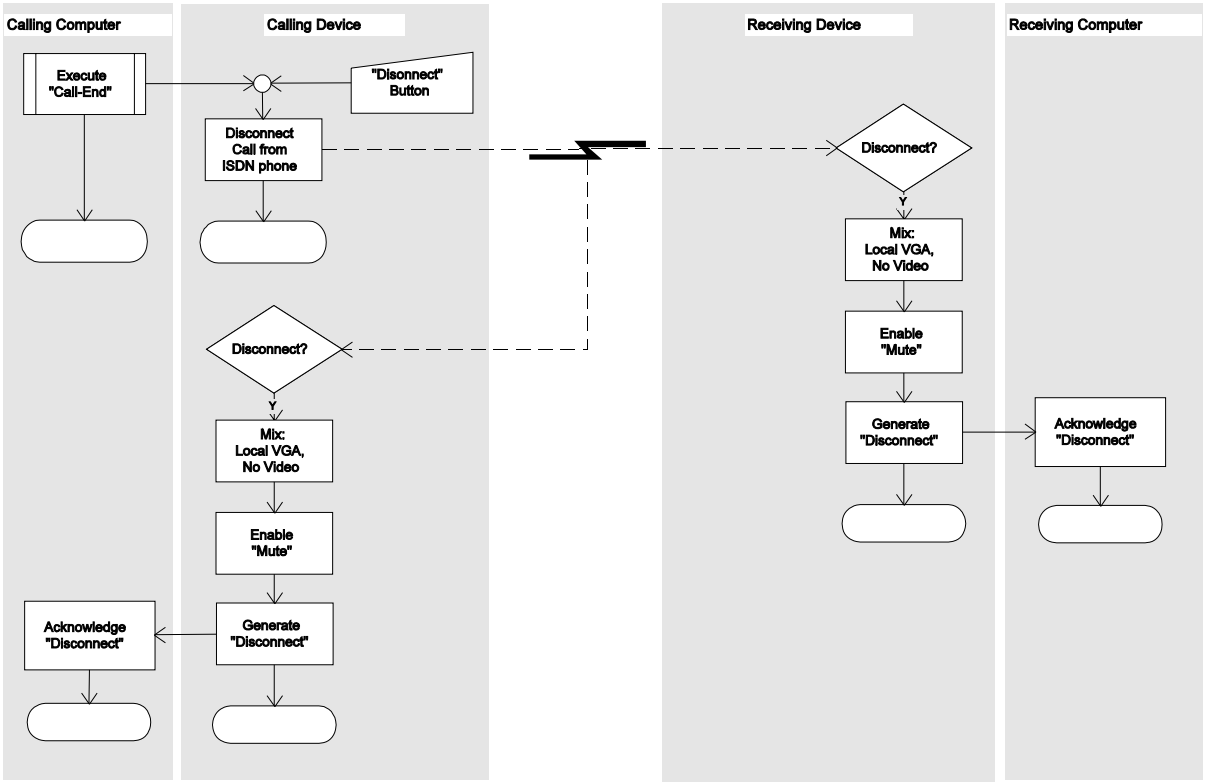
At this point, the BOX then waits for a disconnection to be detected.

2.5 Communication Disconnection Detected:

While the BOX is connected to a remote destination, it will detect when a connection has been broken. Upon detecting the disconnection, the BOX will reset itself to default mode -- i.e. stop accepting any remote audio/video, SVGA, or computer input signals from the remote connection, and set the output mix to the local SVGA only. Also, the BOX will generate an IOCTL command representing "Call-Disconnect" and send it to the local computer. This will notify any device driver software running that a disconnection has occurred.

At this point, the call has ended. An overview of this logic is found below in **Figure 5**.

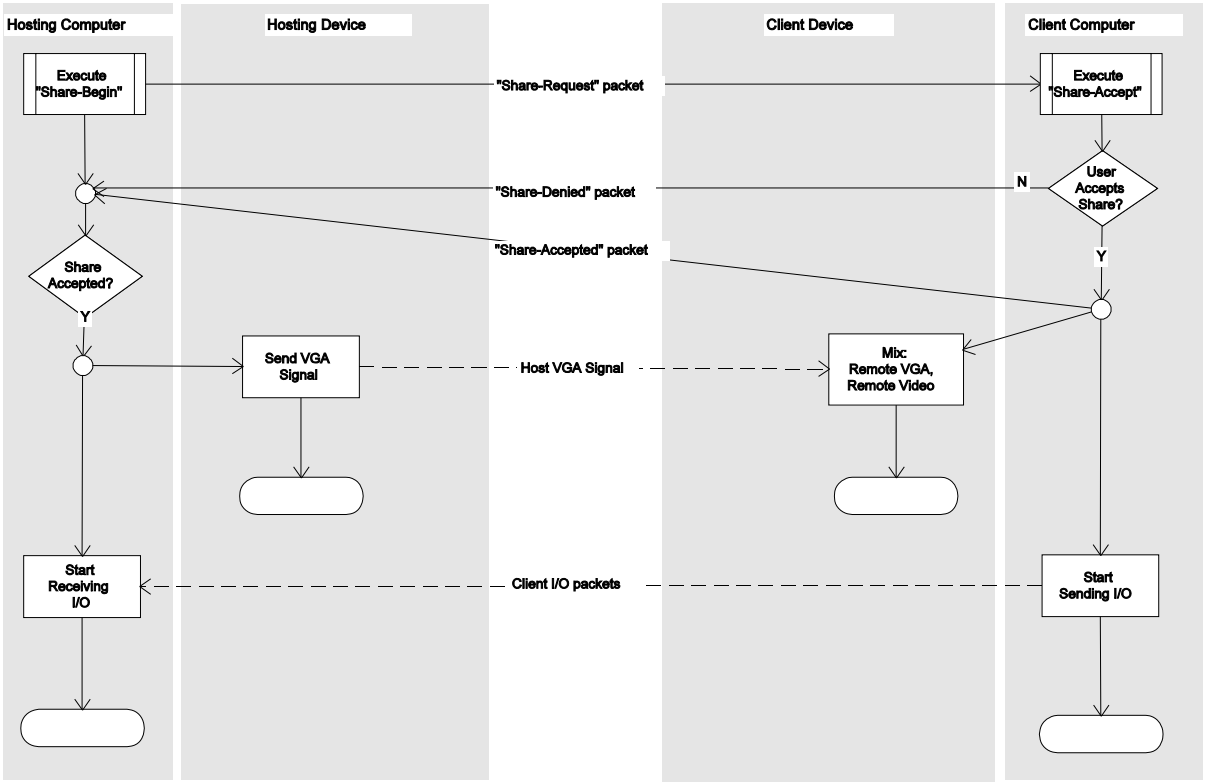
Figure 5 Call-Termination Flow



2.6 Initiate Desktop-Sharing Mode:

The BOX, by default, will mix any incoming audio/video signals with the local SVGA signal. All desktop sharing "modes" are driven by the controlling software in each box. There are two modes for desktop sharing: "Host" and "Client" mode. A host projects its desktop image, and receives input from all clients. Each client receives and displays the desktop image from the host, and redirects all its local computer input to the host. Since these modes are heavily software controlled, all commands to the BOX will be from the local computer (software drivers). A visual overview of the flow required to initiate desktop sharing is found in **Figure 6**.

Figure 6 Desktop-Sharing Initialization Flow



2.6.1 Host Mode:

Upon receiving an IOCTL command representing "Host-Share" from the local computer, the local SVGA signal will be sent and encoded to the remote connection. Since the software drivers will control all computer input (i.e. keyboard and mouse), no other hardware specifications need to be made at this point. Refer to **Figure 3** for an internal view of the BOX during host mode.

2.6.2 Client Mode:

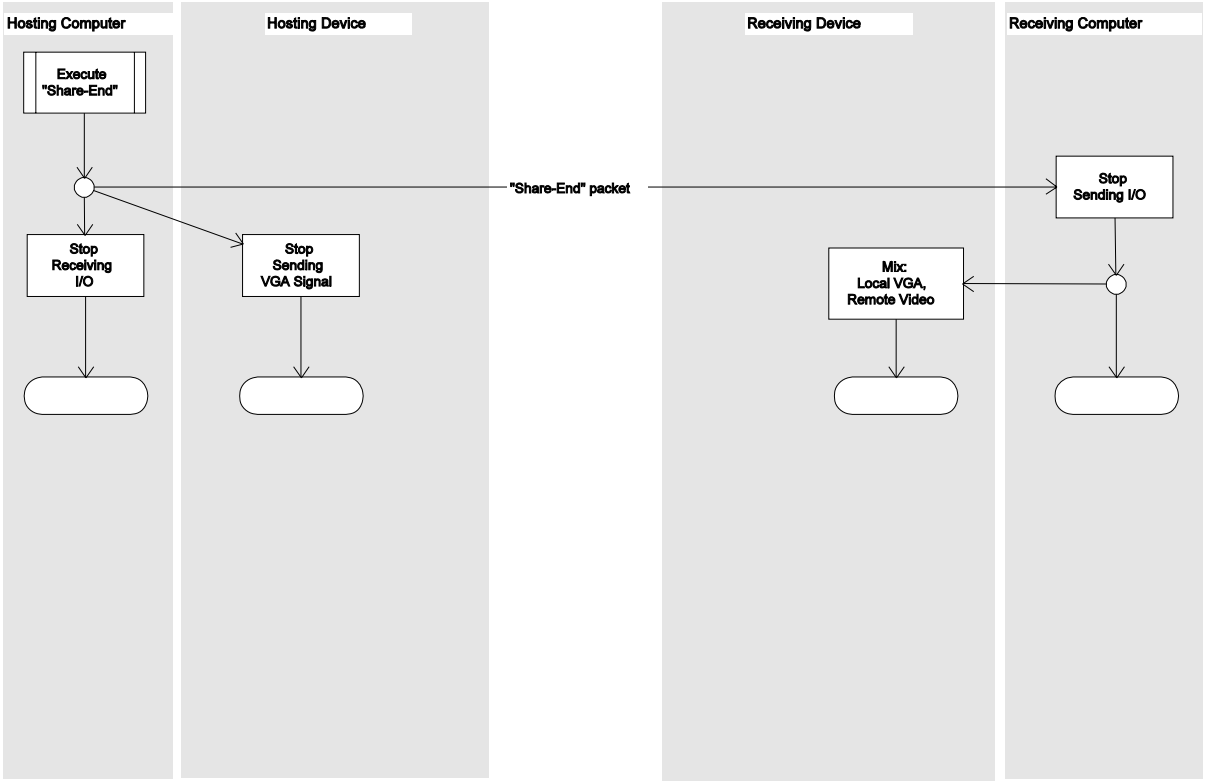
Upon receiving an IOCTL command representing "Client-Share" from the local computer, an SVGA signal will be receive and decoded from the remote connection. At this point, the hardware will also change the SVGA output to mix any incoming

audio/video signals with the incoming SVGA signal. Since the software drivers will control all computer input (i.e. keyboard and mouse), no other hardware specifications need to be made at this point. Refer to **Figure 2** for an internal view of the BOX during client mode.

2.7 Terminate Desktop-Sharing Mode:

Upon receiving an IOCTL command representing "Share-End", the BOX will shut down the SVGA transport channel -- i.e. if in host mode then stop sending SVGA, and if in client mode then stop receiving SVGA. At this time the BOX will reset the mix to remote video and local SVGA signals. A visual overview of the flow required to terminate desktop sharing is found in **Figure 7**.

Figure 7 Desktop-Sharing Termination Flow



3 Software Logic Flow:

The supporting software for this CSCW device is explicitly not defined here. With the capabilities of the proposed hardware, software can be developed that implements different methods of interaction (turn-taking versus free-for-all), different initial states (windowed versus overlayed camera video), and different multi-point capabilities.

4 I/O Control Support

The hardware device must allow I/O Control (IOCTL) support for communication with the local computer, communications device and remote connection device. The local computer will use a serial RS232C port and operate in *null-modem* fashion. The communications device will most likely be an OEM HDSL product (i.e. PairGain, BrookTree, AT&T Paradyne). This CSCW device may connect to either another CSCW device of this type or a standard H.320-compliant device. If the remote connection is a standard H.320-compliant device, only video-conferencing capabilities will be available. If the remote connection is this same type of CSCW device, a hardware handshake command will acknowledge it, and additional desktop-sharing capabilities will be available.

4.1 Software Command Interface

Specific command formats and returned data formats have been specified to provide for consistent data communications between various manufacturers of devices.

For this CSCW device, all commands should be between `0x80` and `0x9F` inclusive (32 commands), all positive return codes should be between `0xA0` and `0xBF` (32 positive return codes), all negative return codes should be between `0xC0` and `0xEF` (48 negative return codes), and special delimiting codes should be between `0xF0` and `0xFF` (16 special delimiters).

All packets should be wrapped by ascii byte-codes representing the start and end of an information packet.

```
"Start-Of-Packet"   0xFF

"End-Of-Packet"   0xFE
```

Although most return packets are defined by the individual command, some standard empty packets are pre-defined. These include:

```
"Command-Acknowledged"        0xA0

"No-Connection"             0xEF

"Invalid-Command"             0xE0
```

```
"Invalid-Command-Parameters"  0xE1

"Data-Send-Failed"          0xE2

"Data-Receive-Failed"       0xE3
```

4.1.1 Command Format

Following each command should be a number representing the number of bytes to follow, excluding the "end-of-packet" character. The command packet would look like:

```
BYTE          "Start-of-Packet" Code

BYTE          Command Code

UNSIGNED LONG  Command Parameter Length

BYTE ...      Command Parameter Byte Stream

BYTE          "End-of-Packet" Code
```

The byte-ordering for any multi-byte codes should be MSB (most-significant-byte) first. If a command is received by the CSCW device where the "end-of-packet" code does not follow the n-th byte specified by the packet, an empty data packet with a status of "bad-data-received" is returned.

4.1.2 Returned Data Format

All return data should be wrapped by the same ascii byte-codes representing the start and end of an information packet. Following each return status should be a number representing the number of bytes to follow, excluding the "end-of-packet" character. The returned data packet would look like:

```
BYTE          "Start-of-Packet" Code

BYTE          Return Status

UNSIGNED LONG  Returned Data Length

BYTE ...      Returned Data Byte Stream

BYTE          "End-of-Packet" Code
```

As with the command format specified previously, the byte-ordering for any multi-byte returned data should be MSB (most-significant-byte) first.

4.2 IOCTL Command Set

These are commands that are sent from within the local computer to the CSCW device to control the device via the software driver.

4.2.1 "Reset"

Returns all CSCW device settings to their powered-on state.

Command Code: `0x80`

Input Parameters: (Byte) Type of reset to perform, which can be any of the following values:

"Soft Reset"	(`0x00`)	Reset device to last settings saved in EEPROM.
"Hard Reset"	(`0x01`)	Reset device to default power-up settings (refer to "Default Device States" below).
"Destructive Reset"	(`0xFF`)	Erase EEPROM settings, and reset device to default power-up settings (refer to "Default Device States" below).

Return Codes: "Command-Acknowledged" (`0xA0`),

"Invalid-Command-Parameters" (`0xE1`).

4.2.2 "Connect"

This command simulates the pressing of the "Connect" button on the CSCW device. The CSCW device processing of the "Connect" button allows it 2 possible functions:

Initiate Call: If there is no connection currently active, then the CSCW device is placed in an internal state that is used to initiate a call to a remote location either by the user manually dialing or the software sending a series of "Comm Data Send" commands (described later). After a connection has been detected, this CSCW device sends out a "CSCW-Handshake" synchronization command. If the CSCW device receives a connection via the "CSCW-Handshake" synchronization command or some other H.320 standard acknowledgement, it starts

encoding/sending local video and audio to the remote connection (basically to identify the caller to the remote location before the call is accepted).

If the acknowledgement was from an H.320 device, the CSCW device then: (1) responds to the H.320 device with an appropriate H.320 acknowledgement code, (2) starts receiving/decoding remote video and audio from the remote connection, and (3) sets the mix to local SVGA and remote video.

Otherwise, if the acknowledgement was from a similar CSCW device, the CSCW device then **waits** for an "Call-Accepted" or a disconnection from the remote connection. If the returned status is positive, the device: (1) starts receiving/decoding remote video and audio from the remote connection, and (2) sets the mix to local SVGA and remote video.

Accept Call: If a connection is already active, the CSCW device has already starting receiving/decoding remote video and audio from the caller (for identification of the caller as described above), and merely uses the "Connect" button processing to accept the call. This processing includes: (1) generating a "Call-Accepted" synchronization command (described later) to the remote location, and (2) encoding/sending local video and audio to the remote connection (to complete the connection to the remote location).

Command Code: `0x81`

Input Parameters: (None.)

Return Codes: "Command-Acknowledged" (`0xA0`),

"Invalid-Command-Parameters" (`0xE1`).

4.2.3 "Disconnect"

This command simulates the pressing of the "Disconnect" button on the CSCW device. Its only function is to terminate the connection. from that point, the CSCW device will sense the line disconnect, and return the mix to local SVGA only.

Command Code: `0x82`

Input Parameters: (None).

Return Codes: "Call-Disconnect" (0xA4),

"No-Connection" (0xEF).

4.2.4 "Set Share Mode"

Defines a desktop-sharing mode used by the CSCW device.

Command Code: 0x83

Input Parameters: (Byte) Sharing Mode to Set, containing one of the following values:

"No share":	(0x00)	Stop all sending/receiving of SVGA signals to/from remote connections. The CSCW device responds to this command by also setting the mix to remote video and local SVGA.
"Host Mode":	(0x01)	Start encoding/sending local SVGA to the remote connection. The CSCW device responds to this command by also setting the mix to remote video and local SVGA.
"Client Mode":	(0x02)	Start receiving/decoding remote SVGA from remote connection. The CSCW device responds to this command by also setting the mix to remote video and remote SVGA.

Return Codes: "Command-Acknowledged" (0xA0),

"No-Connection" (0xEF),

"Invalid-Command-Parameters" (0xE1).

4.2.5 "Query Connection"

Queries the CSCW device of its connection status.

Command Code: 0x84

Input Parameters: (None).

Return Codes: "Connection-Active" (0xA5)

(Byte) Specific sharing mode active as described in the "Set Share Mode" command.

"No-Connection" (0xEF).

4.2.6 "Set Fade Level"

Defines the video/SVGA fade level used by the CSCW device.

Command Code: 0x85

Input Parameters: (Byte) Value ranging from -100 to 100, where -100 displays only the SVGA signal with no remote video mixed in and 100 displays only the remote video signal with no SVGA mixed in.

Return Code: "Command-Acknowledged" (0xA0),

"Invalid-Command-Parameters" (0xE1).

4.2.7 "Query Fade Level"

Queries the video/SVGA fade level used by the CSCW device.

Command Code: 0x86

Input Parameters: (None).

Return Codes: "Command-Acknowledged" (0xA0)

(Byte) A value representing the current fade level, as described in the "Set Fade Level" command.

4.2.8 "Set Video Position"

Defines the placement of video origin with respect to the SVGA screen output by the CSCW device.

Command Code: `0x87`

Input Parameters: (2 WORDS) A pair of integral (X,Y) coordinates in tenths of a percent (i.e. from 0 to 1000) representing the where the top-left corner or the video window is placed relative to the top-left corner of the output SVGA screen.

Return Codes: "Command-Acknowledged" (`0xA0`),

"Invalid-Command-Parameters" (`0xE1`).

4.2.9 "Query Video Position"

Queries the placement of video origin with respect to the SVGA screen output by the CSCW device.

Command Code: `0x88`

Input Parameters: (None).

Return Codes: "Command-Acknowledged" (`0xA0`),

(2 WORDS) A pair of integral (X,Y) coordinates defining the video position as specified in "Set Video Position" Command.

4.2.10 "Set Video Size"

Defines the height and width of the video output with respect to the SVGA screen output by the CSCW device.

Command Code: `0x89`

Input Parameters: (2 WORDS) A pair of integral (X,Y) coordinates in tenths of a percent (i.e. from 0 to 1000) representing the width and height, respectively, of the video output relative to the size of the output SVGA screen.

Return Codes: "Command-Acknowledged" (`0xA0`),

"Invalid-Command-Parameters" (0xE1).

4.2.11 "Query Video Size"

Queries the video size and position used by the CSCW device.

Command Code: 0x8A

Input Parameters: (None).

Return Codes: "Command-Acknowledged" (0xA0),

(2 WORDS) A pair of integral (X,Y) coordinates defining the video width and height as specified in "Set Video Size" Command.

4.2.12 "Set Mute"

Defines the video/audio muting (inhibiting local audio/video from being sent to the remote connection).

Command Code: 0x8B

Input Parameters: (2 Bytes)

Byte 1: The first byte represents a bit-mask of elements to mute. The bit-masks are defined as follows:

Video Mute: (00000001) Do not send local video signal to remote connection.

Audio Mute: (00000010) Do not send local audio to remote connection.

Host Mute: (00010000) Only applicable while CSCW device is in host mode, temporarily suspend sending desktop to remote connection.

Client Mute: (00100000) Only applicable while CSCW device is in client mode, temporarily suspend receiving host desktop (and mix incoming video with local SVGA).

Byte 2: The second byte represents the masking action to use. These are defined as follows:

Replace Bits: (0x00) Replaces all the mute bits with the bit-mask passed.

Enable Bits: (0x01) Turns on bits that are specified by the bit-mask.

Disable Bits: (0x02) Turns off any bits that are specified by the bit-mask.

Return Codes: "Command-Acknowledged" (0xA0),

"Invalid-Command-Parameters" (0xE1).

4.2.13 "Query Mute"

Queries the video/audio muting (inhibiting local audio/video from being sent to the remote connection).

Command Code: 0x8C

Input Parameters: (None).

Return Codes: "Command-Acknowledged" (0xA0)

(Byte) A value representing the current mute masks enabled, as described by Byte 1 of the "Set Mute" command.

4.2.14 "Set Video Mix"

Defines the video/SVGA signal mixing used by the CSCW device. The bits are determined such that passing an all zero bit mask will result in the default mode when the CSCW device is reset.

Command Code: 0x8D

Input Parameters: (2 Bytes)

Byte 1: The first representing a bit-mask of elements to mix. They are defined as follows:

VGA Mix: (00000001) Boolean specifying whther to use remote SVGA (=1) or local SVGA (=0).

VGA Off: (00000010) Boolean specifying whether to bypass any SVGA signals (=1) This overrides any "VGA Mix" bit setting.

Video Mix: (00000100) Boolean specifying whether to use the remote video (=0) or local video (=1) in the mix.

Video Off: (00001000) Boolean specifying whether to bypass any video signals (=1) This overrides any "Video Mix" bit setting.

Reserved: (11110000) These bits will be used later on to have the CSCW device automatically display preset video messages on the user's monitor.

Byte 2: The second byte represents the masking action to use. These are defined as follows:

Replace Bits: (0x00) Replaces all the mute bits with the bit-mask passed.

Enable Bits: (0x01) Turns on bits that are specified by the bit-mask.

Disable Bits: (0x02) Turns off any bits that are specified by the bit-mask.

Return Codes: "Command-Acknowledged" (0xA0),

"Invalid-Command-Parameters" (0xE1).

4.2.15 "Query Video Mix"

Queries the video/SVGA signal mixing used by the CSCW device.

Command Code: 0x8E

Input Parameters: (None).

Return Codes: "Command-Acknowledged" (0xA0)

(Byte) A value representing the current mute masks enabled, as described by Byte 1 of the "Set Mute" command.

4.2.16 Store Settings

Saves the current CSCW device settings for video position, size and fade level to an EEPROM.

Command Code: `0x8F`

Input Parameters: (None).

Return Codes: "Command-Acknowledged" (0xA0)

4.2.17 "Remote Data Send"

Allows software to talk to the remote connection. The CSCW device strips off all the delimiting data as well as the command code, and packetizes and passes the variable binary data to the remote connection (if possible, by using Out-Of-Band channels). If the remote connection returns data, it is delimited and a command code of "Remote Data Received" is generated and sent to the local computer (described below).

Command Code: `0x90`

Input Parameters: (Variable) This is a binary packet that includes data specific to the software drivers at the remote connection.

Return Codes: "Command-Acknowledged" (`0xA0`),

"Data-Send-Failed" (`0xE2`),

"Invalid-Command" (`0xE0`),

"Invalid-Command-Parameters" (`0xE1`).

4.2.18 "Comm Data Send"

Allows software to talk directly to the communications device. The CSCW device strips off all the delimiting data as well as the command code, and passes the variable binary data directly to the device. If the device returns data, it is delimited and a command code of "Comm Data Received" is generated and sent to the local computer (described below).

Command Code: `0x91`

Input Parameters: (Variable) This is a binary packet that includes data specific to the local communications device.

Return Codes: "Command-Acknowledged" (0xA0),

"Data-Send-Failed" (0xE2),

"Invalid-Command" (0xE0),

"Invalid-Command-Parameters" (0xE1).

4.3 Hardware-Generated Notification Commands

These are commands that are sent from within the local CSCW device to the local computer as notification of events such as data received from a remote connection.

4.3.1 "Remote Data Received"

After receiving packetized controlling data from the communications device, the CSCW device should generate this command to pass this data to the local computer.

Command Code: 0x92

Input Parameters: (Variable) This is a binary packet that includes data specific to the software drivers at the remote connection.

Return Codes: "Command-Acknowledged" (0xA0),

"Data-Receive-Failed" (0xE3),

"Invalid-Command" (0xE0),

"Invalid-Command-Parameters" (0xE1).

4.3.2 "Comm Data Received"

After receiving data via the communications device, the CSCW device should generate this command to pass this data to the local computer.

Command Code: 0x93

Parameters: (Variable) This is a binary packet that includes data specific to the local communications device.

4.3.3 "Line-Connected"

After sensing a connection via the communications device, the CSCW device sends a "CSCW-Handshake" synchronization command followed by some standard H.320 standard handshake acknowledgement to the remote connection. If it first receives either a "CSCW-Handshake" synchronization command or some standard H.320 handshake acknowledgement, there are two possible processing trees:

(1) If this CSCW device originally initiated the call via "Connect" button processing, this device will start encoding/sending local video and audio to the remote connection, and *no "Line-Connected" message is generated*.

(2) If this CSCW device was originally idle, this device will start receiving/decoding remote video and audio, set the mix to local SVGA and remote video, and *a "Line-Connected" message is generated*.

Command Code: `0x94`

Parameters: (None).

4.3.4 "Disconnect-Sensed"

After sensing disconnection of the communications device, the CSCW device first sets the mix to local SVGA only, and generates this command to pass to the local computer.

Command Code: `0x95`

Parameters: (None).

4.4 Hardware-Generated Synchronization Commands

These are commands that are sent from within the local CSCW device to a remote CSCW device.

4.4.1 "Call-Accepted"

This command can be generated by either software or the CSCW device itself. To review the scenario up to this point: upon receiving a connection, the CSCW device will start mixing local SVGA with remote video signals, and a "Line-Connected" command is sent to the local computer. The software driver, if loaded, will receive the "Line-Connected" message and display a dialog asking the user to accept or refuse the call. The user, at this point, may accept the call by either selecting the appropriate option form the software dialog box, or by pressing the "Connect" button on the CSCW device itself.

(1) If the user accepts the call via the dialog box, the software driver generates the "Simulate-Connect-Button" message to its CSCW device.

(2) If the user presses the "Connect" button after a line has been connected, the CSCW device should generate this command to pass to the remote connection. The remote connection, upon receiving this, will pass this to its local computer.

Command Code: `0x96`

Parameters: (None).

4.4.2 "CSCW-Handshake"

Once a connection is sensed on both sides of the connection, both CSCW devices generate this command to identify themselves. Each then waits for this same command in return; thus confirming the handshake.

Command Code: `0x9F`

Parameters: (None).

4.5 Default Device States

At power-up, the CSCW device will read any settings stored in its EEPROM. For any settings that do not exist, the following initial states are to be set:

Fade Level: The initial fade level will equally mix video and SVGA signals (0).

Video Position: The initial video position is set to the screen origin (0,0).

Video Size: The initial video size is set to full-screen (1000,1000).

5 Hardware Acceptance

The following issues have been identified for acceptance of the proposed CSCW device. These are considered minimal requirements, and may change as the project progresses. Where possible, video-conferencing standard compression should be used for compatibility.

Physically, this device must be a portable unit (< 15 lbs.) that can be ideally placed within reach of a user's computer monitor, camera, spekaers and microphone.

Technically, this device should be able to have video-conferencing capabilities with any H.320-compliant hardware. Due to the novel method of desktop-sharing, it will preclude any desktop- or application-sharing capabilities with other types of hardware.

5.1 Audio Transport

Audio transport should have the highest priority. It should be transported independent of video. Quality of audio sampling should minimally be 11kHz monophonic.

5.2 Video Transport

Video transport should have medium priority. It should be digitized NTSC standard rate (512x480) for transport using the video-conferencing standard hardware compression scheme (H.261 ?). Received video should be interpolated to the match the signal rate of the SVGA signal to be displayed. Interpolation should use anti-aliasing (re-sampling) techniques for improved quality. Also, a minimum refresh rate of 15-20 frames per second is required during desktop-sharing mode (i.e. simultaneously transporting SVGA signals), while a refresh rate of 25-30 frames per second is required during video-conferencing mode (i.e. no transport of SVGA signals).

5.2.1 SVGA Transport

SVGA transport (for desktop-sharing mode) should have medium priority. It should be digitized using the most effective hardware compression techniques available to achieve a minimum refresh rate of 5-10 frames per second during desktop-sharing mode.

The following SVGA modes should initially be supported:

Resolution (Columns x Rows)
640 x 350
640 x 400
640 x 480
800 x 600
1024 x 768
1280 x 1024

5.3 Input Control Transport

Input control will be supported by the CSCW device via IOCTL commands. The hardware should accept IOCTL commands representing "Pass-Data" along with an attached data structure, packetize the data, and send it via an Out-Of-Band channel, if possible.

(end of document)