THE AUDIO VIDEO SERVER ATTENDANT: INCREASING ACCESS AND CONTROL OF MEDIA SPACES

by

Anuj Uday Gujar

A thesis submitted in conformity with the requirements
for the degree of Master of Science
Graduate Department of Computer Science
University of Toronto

THE AUDIO VIDEO SERVER ATTENDANT: INCREASING ACCESS AND CONTROL OF MEDIA SPACES

Masters of Science 1996
Anuj Uday Gujar
Department of Computer Science
University of Toronto

Abstract

A media space is an environment which integrates audio, video and computer technologies in novel ways to provide its members with new ways of interacting and working together. Person-to-person communication to a media space from outside the media space is limited. This thesis presents the user-centered, iterative evolution of the audio video server attendant (AVSA)- a system that enables people to *access and control* the resources of a media space *from a conventional videoconferencing room*.

A key issue is that the only equipment available at the videoconferencing room is the traditional videoconferencing equipment- a camera, monitor, speaker and microphone. Consequently, we are forced to enable control to be exercised through speech input in response to visual prompts.

In a larger context, we are seeing a rapid move towards networked interactive information appliances. Conventional efforts "converge" the telephone, television and/or computer technologies to produce these appliances. However, because of the approach, the shape that the associated services are assuming is rooted in the appliances of the past. As an example of a networked interactive information appliance not centered on any one technology, the AVSA presents an option to the limited perspective.

Acknowledgments

I began work on this thesis in March of 1995. Over the past year and a half many people have contributed to its completion. Above all I thank my supervisor and friend, Professor William Buxton who introduced me to the world of HCI and provided me with the opportunity to investigate my curiosities. I also thank my second advisor, Dr. Ronald Baecker, for taking the time from a busy schedule to provide input and comments on this work. I also thank Dr. Marilyn Mantei for her valuable advice at various times during the course of this work.

I thank Koichiro Tanikoshi, a visiting scientist from the Hitachi Research Laboratory in Japan, who started the HyperDoorway project providing a base from which the AVSA could be developed.

A special thanks to Tracy Narine for his efforts in helping me integrate this work into the University of Toronto environment. I thank Victor Ng-Thow-Hing, Kevin Schlueter, Alex Mitchell, Blair Andrew McLeod and Petros Faloutsos for their superb system administration without which this work would have been impossible. I also acknowledge Jeremy Cooperstock for permitting me to integrate this work with the reactive room project.

I thank my family for their unquestioned and unrelenting support over the past two years.

I thank DAWGmasters Jin Lee and Ian Maione and my buddy Andrea Leganchuck for providing hours entertainment without which I could not have completed my masters.

I also thank the DGP lab, IRG group, University of Toronto and the members of each organization for their input, advise and the use of resources.

I thank the Natural Sciences and Engineering Research Council for supporting me financially during the course of this work. Finally, I thank Alias | Wavefront Inc. and the supporting members of the Ontario Telepresence Project for providing the resources necessary for this research.

For my father, my mother and my sister.

Table of Contents

List of Figures

CHAPTER 1 Introduction

A media space is an environment which integrates audio, video and computer technologies in novel ways to provide its members with new ways of interacting and working together[20]. Unfortunately, communication to a member of a media space is limited to people within the media space and to people who belong to another media space. The number of media spaces available is very small and due to the cost associated with media spaces the number will most likely remain small in comparison to the number of traditional videoconferencing rooms.

The AVSA is a system that significantly increases the number of people who can communicate to members of a media space by providing an interface through which people with access to a traditional videoconferencing room can access or *visit* the media space. In addition, by providing control over media space resources, the AVSA enhances the effectiveness of the visit.

1.1 The "Convergence" Theme

Consider the airplane, the telephone, the television, the computer. What do all of these technologies have in common? In terms of communication, they are all society's attempts to make it easier and more efficient to exchange information. Each of these technologies have properties that contribute to their purpose as well as properties that detract from their purpose.

The airplane, along with many other technologies, allows one to travel quickly from point A to point B so that information can be exchanged in a highly interactive and effective manner (face-to-face communication). Unfortunately, travelling to a location is not always a viable option. There may be political situations, time restrictions or cost limitations that make travelling impossible. A more complex situation is one where a person at point A must meet with a person at point B and C. In this situation A must make a choice as to who to have the face-to-face communication with. Face-to-face with B means remote communication with C and vice versa.

The telephone addresses the limitations of technologies like the airplane by enabling one to instantly communicate with virtually anyone else in the world at a relatively low cost and at any time two parties are willing. The drawback is in the level of interaction. Without the visual cues and other advantages of face-to-face communication it becomes difficult to convey ideas. It is also difficult to participate in three-way or more communication.

The television allows communication to groups of any size at any time. The level of communication is also relatively high as it integrates high quality audio and video. The problem with this technology is that interaction stops at the remote control. One has no other means of choosing what information to receive and when to receive it aside from using the schedule of the television networks. A more complex system, such as one augmented with a programmable VCR, would allow one to organize their own programming schedule, but this requires significant planning. In any case, this setup does not allow one to contribute ones own thoughts and ideas. Thus, the interactivity is not symmetrical.

Computers enable a high degree of communication at a very low operating cost. One can communicate using audio, video, text or any combination of the three. One also has the ability to access information on virtually any topic at any time. The most well known method for achieving this is through the internet. There are many problems with using this technology. First, a person must have access to the technology and second they must learn how to use it. A third problem is that many times the experience of using the technology is significantly diminished because the technology is too visible.

A further problem with computer-based communication can be attributed to the limited bandwidth available at any point in time. Consider talk programs. Talk programs facilitate communication through text. Text does not require much bandwidth. Therefore, performance of the system is quite good. However, text is not very expressive and thus is not a very effective way to communicate. As a result, the quality of the interaction is low. By using a more expressive medium or, equivalently, richer media we can theoretically increase the quality of the interaction. The problem is that the richer media requires more communications bandwidth. For example, web-phones and internet-phones use audio to enable a person to communicate. A common complaint of the users is that either the quality of the audio is poor or the audio is "choppy". This reduced performance is a direct result of the limited bandwidth. Consequently, the quality of interaction may be only slightly increased. Furthermore, systems which incorporate audio and video should theoretically have the highest quality of interaction. This is not the case. In fact, when there is a lot of traffic over the network the performance is reduced so much that the quality of interaction may actually lower than that of simple talk programs. In summary, due to bandwidth limitations using more expressive media does not necessarily translate to an increase in the performance of the system and thus can compromises the quality of interaction.

By and large, the developers of the computer, television and telephone technologies are attempting to address the deficiencies outlined above by developing appliances which mimic capabilities of other technologies that do not have the deficiencies. However, because of the limitations within which each technology must operate, they will only succeed to a certain degree. Unfortunately, as these and other technologies develop along their own paths our lives are being complicated by appliances which, although fewer in number, attempt to offer more than the technology which they are applying can realistically support. Furthermore, because of the approach, these appliances typically function as technology islands, as opposed to a suite, working in concert.

As described earlier, the common goal is to enable the efficient and effective exchange of information. For a number of years it has been speculated that the proper way to achieve this goal is to develop appliances which *appropriately* converge different technologies, exploiting the advantages of each and reducing the impact of the disadvantages of each. The work presented in this thesis describes the development of an example of one such appliance called the Audio Video Server Attendant (AVSA).

1.2 The Key Constraints

The approach we take to design and develop the system are influenced by two key constraints:

• no control of the available equipment
• limited time

The first key constraint is that we have no control over what equipment is available at the videoconferencing room. The only thing we can assume is that the videoconferencing room has the traditional videoconferencing equipment- i.e. a camera, a monitor, a microphone and a speaker. This constraint limited our design options in that the AVSA system had to “live” at the media space and, based on our evaluation, be controlled through speech input in response to visual prompts.

The second key constraint is limited time. If this work is to have any influence on the ongoing dialogue among the explorers of convergence, it must be completed in a *timely* manner without sacrificing *usability*. This has clear implications on our approach in developing the system. Instead of spending a lot of time on formal evaluation, we focus on thoughtful, iterative design with frequent informal evaluation. While this may seem less than ideal, literature[18] and the success of the resulting system supports our approach.

1.3 Summary

In summary, the AVSA acts as an example of *appropriately* converging technologies to help expand the limited perspective on the ongoing discourse on converging technologies[14]. In addition, this work solves a real-world problem associated with media spaces by greatly increasing media space accessibility and enabling control of media spaces. This is accomplished by building a *usable* and *useful* system (the AVSA) in a *timely* manner that enables one to access media space resources.

1.4 Thesis Organization

Chapter two introduces the AVSA concept, our development strategy and our evaluation strategy. Chapter three discusses the basic design considerations, based on the technology we will be using, of the AVSA. Chapters four and five describe two successive stages in the development of the AVSA. Chapter six presents some issues that will need to be addressed before deploying the system into the field for use. A summary of conclusions and contributions are also presented at this point.

CHAPTER 2 Background

This chapter presents a description of the status quo in videoconferencing and media space conferencing in order to establish the AVSA concept. We then describe the general development and evaluation strategies used to develop the system.

2.1 The AVSA Concept

The idea for the AVSA was first introduced by Professor William Buxton. His experiences at Xerox-EuroPARC with the Integrated Interactive Intermedia Facility (IIIF)[4] and at the University of Toronto with the Ontario Telepresence Project (OTP)[22] uncovered a serious deficiency in the quality of interaction experienced by electronic visitors to a media space[3]- namely the limited access and control of media spaces. The following sections describe the situation in detail.

2.1.1 Traditional Videoconferencing

In traditional videoconferencing a person at conference room A communicates to a person at conference room B in real-time through the public switched telephone network or, equivalently, PSTN (Figure 1). The connection between points is made just like a regular telephone call. A person at conference room A pre-arranges a meeting at a certain time with a person at conference room B and then dials B’s number (or vice versa).

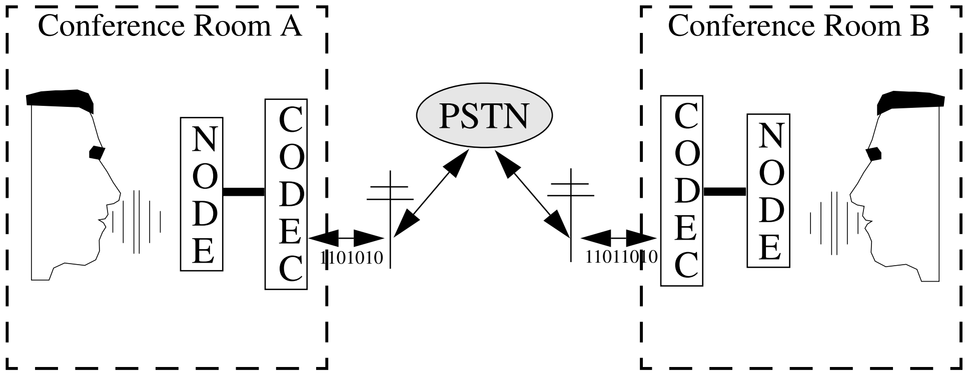


FIGURE 1 The traditional videoconferencing setup.

What makes this connection special is the media rich nature of the communication, i.e. high quality audio and video (A/V). Both parties are equipped with what will henceforth be referred to as a node. A node (Figure 2) consists of a microphone, speaker, camera and monitor.

FIGURE 2 This picture shows the camera, monitor, microphone and speaker of a node.

The A/V obtained at each conference room must to be transmitted to the other in real-time. To accomplish the transfer of this high-bandwidth analog data, each conference site is also equipped with a coder/decoder (CODEC). The CODEC quickly converts the analog A/V data into digital data and transmits it to the other site through a high bandwidth digital PSTN subscriber line. This node/CODEC setup of the traditional videoconferencing room is often called an orphan CODEC.

2.1.2 Media Spaces

Functionally, a media space is an extension of the traditional videoconferencing system in a local setting. When a person at node A wants to connect to a person at node B, the person at node A schedules the meeting and then calls node B. The main difference lies in the architecture (Figure 3) and, consequently, the way the connections are actually made.

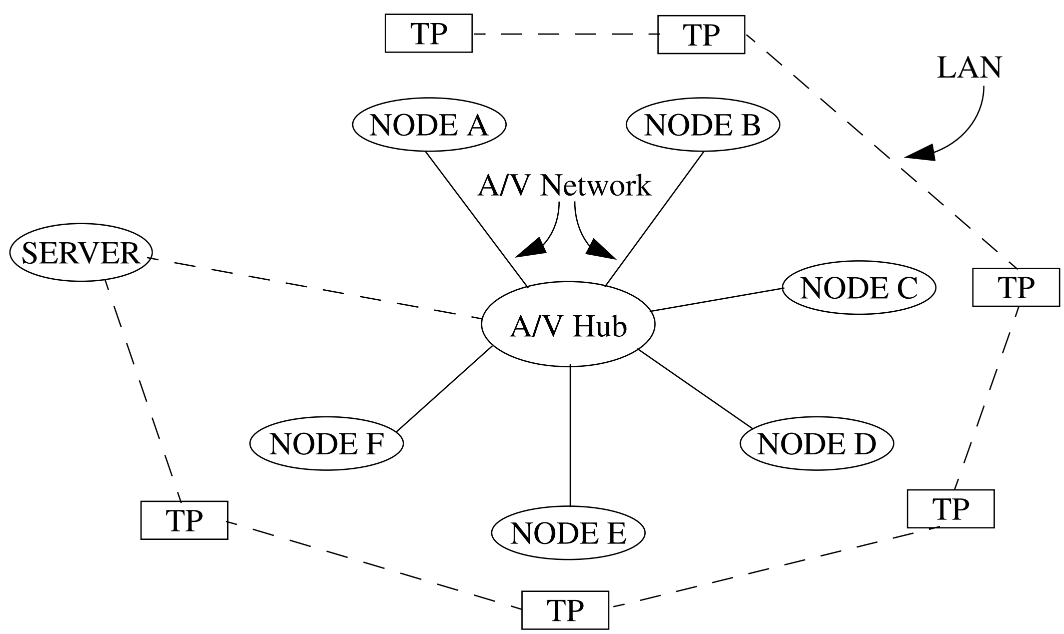


FIGURE 3 A media space allows videoconferencing through a GUI controlled LAN and local A/V network.

In traditional videoconferencing the A/V information is routed though the PSTN. In the media space environment A/V information is routed through a local A/V network called the hub. At the University of Toronto this hub is a hardware/software system called the IIIF.

In traditional videoconferencing the actual routing is also performed by the PSTN. The destination is specified by dialing the number of another CODEC. The PSTN then completes the connection between the two CODECs. Dialing a CODEC is the same as dialing a room because each CODEC is associated with a specific room. In the media space environment the routing is handled by a server connected to the local area network (LAN).

To complete a connection the following events must occur. First, the caller accesses a graphical user interface (GUI), called the telepresence (TP) application, on a computer associated with a node. This computer is connected to the LAN. Then, using the TP application, the caller selects a node (which is essentially a room) to connect to. The TP application sends this request to the server which in turn sends the appropriate commands to the IIIF specifying how the A/V signals should be routed. The IIIF then takes over and makes the appropriate A/V connections.

The first advantage of the media space is that it gives a person a choice as to where or from where to meet with any one of a number of people. A second advantage is that the nature of the local A/V network yields high-quality A/V connections that can be executed quickly when compared to traditional videoconferencing.

The disadvantage is that it only allows communication between people who are connected to the local A/V network. No provision is made to contact people outside the local system. To address this deficiency we add a CODEC to the media space and update the architecture (Figure 4). The procedure for making a connections within the media space remains the same. However, to make a connection to a traditional videoconferencing site two things must occur. First, the local node must connect to the local CODEC. Second, the local CODEC must connect to the remote CODEC. Both events are facilitated through the TP application available to the local node. The user uses the TP to tell the server, through the LAN, to make an A/V connection between the local node and the local CODEC. The user then uses an extension of the TP to tell the server to tell the local CODEC what number to dial. The local CODEC then dials the number of the remote CODEC. Communication can commence once someone at the remote CODEC answers the call.

Although long distance communication is enabled with this setup we still find the system deficient in that there is no provision for a member of one media space to communicate with a member of another media space. It is possible for a member of one media space to call the other media space CODEC to CODEC, but without the servers of the two media spaces to negotiate the connection of the nodes the call is useless.

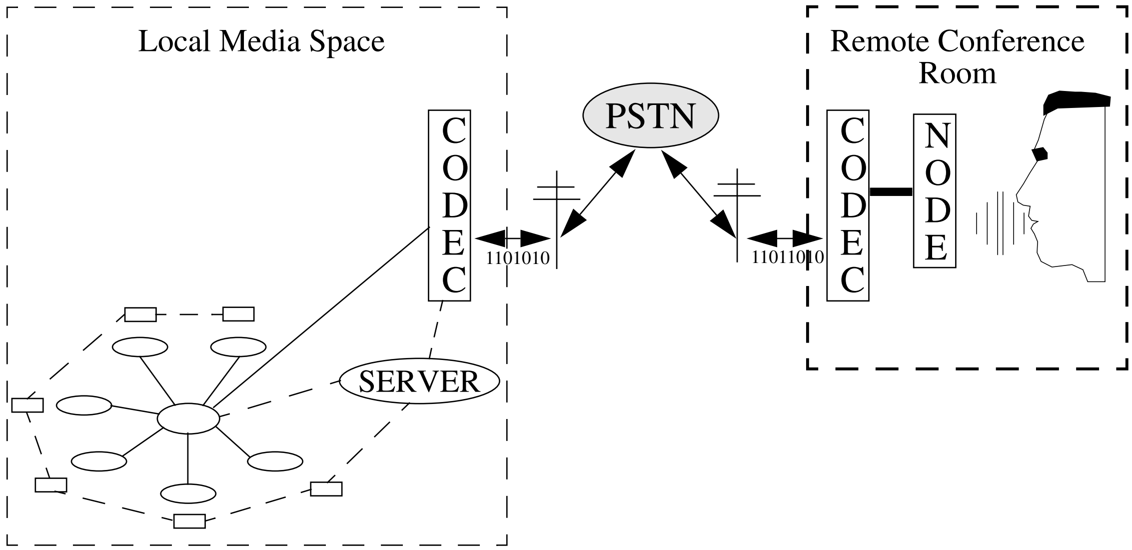


FIGURE 4 Media space connecting to a traditional videoconference room.

In order to make a person to person connection through two media spaces the following three connections must be made. The local node must connect to the local CODEC. The local CODEC must call the remote CODEC. The remote CODEC must connect to the remote node. To address this problem IIIF-2-IIIF communication was developed (Figure 5). Signalling is handled by the local server. When the local node requests to connect to a remote node the local server negotiates the appropriate connections with the remote server over a wide area network (WAN) or internet.

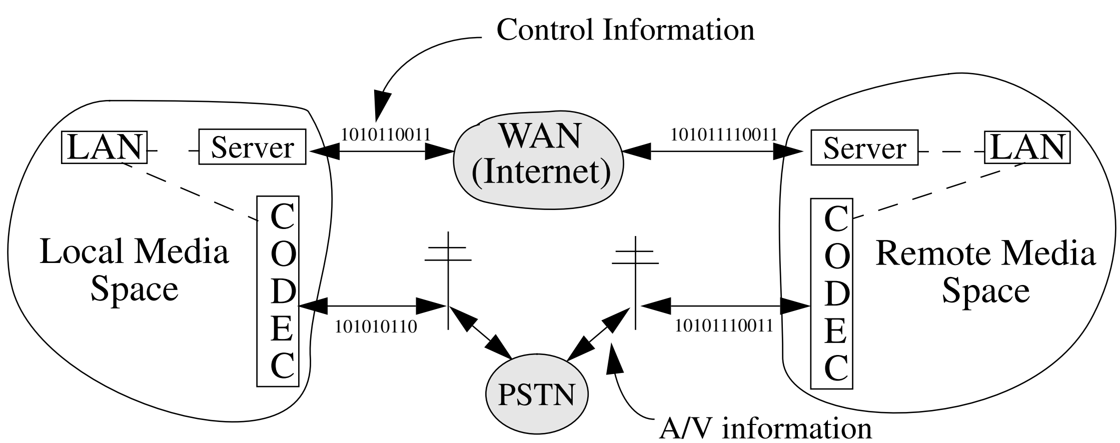


FIGURE 5 Architecture for IIIF-2-IIIF communication.

Unfortunately, there are many more traditional conference rooms than media spaces. Therefore, it is essential that communication between the media space and the traditional videoconference room be provided. We already looked at making a connection *from* a media space *to* the traditional conference room. Now let us look at making a connection from the traditional conference room to a media space (Figure 6). In order to make a person-to-person connection from a traditional videoconference room, two intermediate connections must be made. First the locaexecutedl CODEC must call the remote CODEC. Second, the remote CODEC must connect to the remote node.

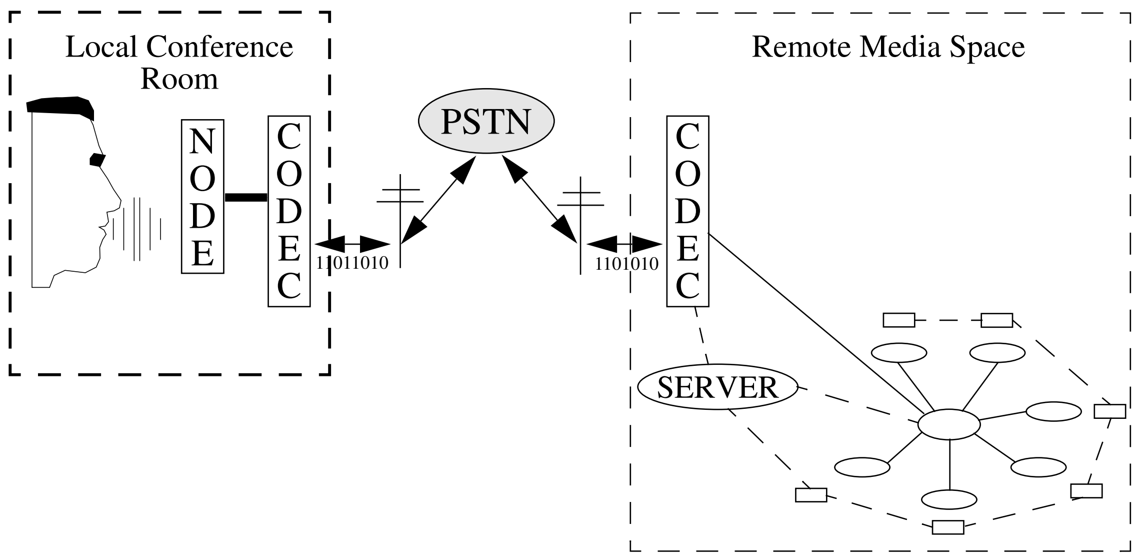


FIGURE 6 Traditional videoconference room connecting to media space.

The first connection can easily be made. However, a traditional videoconferencing room has no control over the server of a remote media space. Therefore, the connection of the remote CODEC to a remote node cannot be made. It is precisely this issue that the AVSA is designed to address.

2.1.3 The Deficiencies

The two deficiencies that stem from the inability of a person at a traditional videoconferencing room to control a remote media space's server are the inability to:

- make a connection to a person
- control media space resources

The following sections discuss these deficiencies further.

Making a connection

To visit the media space without the AVSA, a visitor at an orphan CODEC must go through the following sequence of events:

- pre-arrange a time with a member of the media space
- call the media space's CODEC
- wait to be received by a member of the media space (i.e. wait for the member to connect to the local CODEC through the TP application)

Imagine the analogous situation in telephony (Figure 7). Party A wishes to place a call to party B. A dials B's number. A is connected to a central switch and then must wait for B to dial A's number before the two individuals can talk. This situation is absurd and would be completely unacceptable. However, if it were not for automatic switching systems it would be norm. Unfortunately, it *is* the norm for media space communication from an orphan CODEC to a media space.

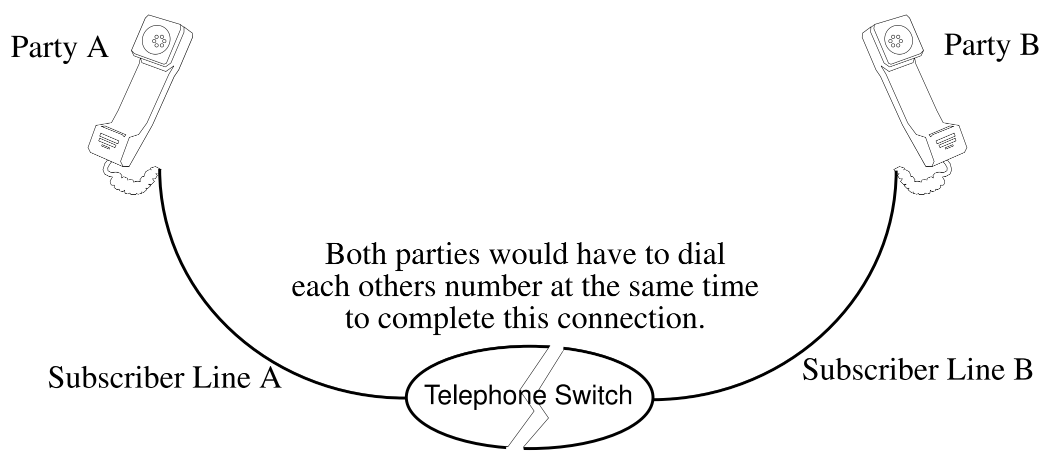


FIGURE 7 The analogous telephony situation without automatic switching.

The root cause of the problems is that the visitor has no control over the media space to which they are connected[15]. The following is a list of problems/consequences associated with this type of setup:

- Most important meetings cannot be planned. They are often ones which involve important business or strategic decisions. By forcing a step which may involve time consuming phenomenons such as “phone-tag” or “e-mail-tag”, the effectiveness of using the system is seriously hampered.
- Once connected to the media space the visitor is stuck waiting to be received. The visitor has no way of knowing when or even if they will be received. It is possible that the person with whom the meeting was arranged was held up and will be late. It is even possible that the person forgot about the meeting!

Controlling media space resources

Lack of access to a media space is not the only problem associated with visiting a media space. One of the goals of media spaces is to allow people to attend/participate in meetings without being physically present, but at a level that would enable communication as though they were physically present. To achieve this it is important for us to ask the question “For effective communication, is it sufficient to provide an audio/ video connection to a media space *or* should the visitor have some degree of control over the media space?” Research[9][10][13] and experience tell us that the latter is more sensible.

Imagine a person electronically joining a meeting that is in progress. However, the camera from which the visitor is supposed to be viewing the conference is blocked by a physical attendee. How can the visitor correct the situation? In the current setup the only way is to ask a physical attendee to switch the visitor to a different camera or to ask the physical attendee to move to a different seat within the room. Both solutions disturb the flow of the meeting and can be irritating to the physical attendees.

There are many other scenarios in which the visitor must ask the physically present person to alter the states of devices for them. Aside from disturbing meetings this process also detracts from the visitors experience of the visit. The visitor feels more like a passive observer (as though they were watching television) instead of an active participant. Thus, by giving control to the visitor their experience will be greatly enhanced.

The final deficiency with the current setup is that there is no way to explore the media space and obtain information without actually talking to a member of the media space. From the visitor’s point of view it is important to be able to access information at any time (as with television technology) and, equally important, to be able to specify what information to access at any time . From the point of view of the members of the media space this issue is important in that one should be able to provide an information bank from which visitors can access information. Once again, we can provide visitors access and control to resources that will enable this type of information exchange.

2.2 Development Strategy

Having defined the problems, the next task is to decide how to develop the system we want in a timely manner, within our budget and without sacrificing usability of the eventual interface. We decided on a three stage iterative technology-driven[8] process.

The first stage is to outline the basic design of the system. First we consider the different technologies, their limitations and their affordances. Then by acquiring input from various individuals familiar with media spaces and individuals not familiar with them we assess the needs of the system and use the technology considerations to produce the basic design of the system.

The next stage is to build a prototype of the system as quickly as possible. With this prototype we:

- will be able to informally evaluate the system and, if all goes well, have proved the concept.
- will be able to obtain user input and further information regarding the interface. This information will be incorporated into the next iteration to improve the system.

The final stage is to build a self-contained system based on the knowledge gained in the first two stages. This system will be transportable to other sites with similar media spaces for further research and/or evaluation.

2.3 Evaluation Strategy

Our goal is to create a usable system in a timely manner. Therefore, no formal evaluation of the AVSA will be done. All evaluation is done in an informal setting on two levels.

The first level of evaluation is on an ongoing basis. This means that as features are implemented they are evaluated. This level of evaluation is performed by the development team and members of the Input Research Group (IRG). The process is to observe or use the interface, looking at specific features, and suggesting ways to improve interaction.

The second level of evaluation will take part at the end of the second development stage. This level of evaluation is based on comments of people from within the DGP lab and people from outside the university. These people will have varying levels of knowledge of computer science and videoconferencing. The process will be to:

- introduce people to the interface
- explain the purpose and its abilities
- ask the person to complete some tasks, like visit a room, play a particular demo, contact a media space member
- observe how they perform the task
- upon completion of the task note their comments on how to improve the interface.

Some people outside the university, who cannot come to the lab to try the system, will be given a written description of the interface and a video tape of a person using the interface and asked to evaluate the system. They detail their general impressions of the system and forward these points to us.

CHAPTER 3

Stage 1: System Considerations

This chapter describes the first stage in the development of the AVSA- defining the basic look and feel of the interface. To accomplish this we first present five qualities that all elements of the system should adhere to. We then consider all of the options available to the interface and, based on the qualities, choose what technology to use for presentation, input and feedback. Finally, we present the general architecture of the system.

3.1 The Qualities

From our observations and input from various users we identified the following five qualities that the AVSA should have:

- it should cater to lowest common denominator
- it should be as ubiquitous as possible
- it should be self-explanatory
- it should work seamlessly in concert with the “privileged” system
- it should be non-intrusive

First the interface should cater to the lowest common denominator (LCD). That is to say that the system should place as few new requirements for equipment and require as few new skills as possible. The system should be functional without adding extra computer or videoconferencing equipment to the orphan CODEC and it should be usable by anyone with aspirations to communicate information.

The second quality is that the system should be as ubiquitous as possible[27]. In our context this means the user should be able to use the system from any node without being attached to any special equipment.

The third quality is that the system should be self-explanatory. This will minimize the training required for new users to learn the system. Note that a self-explanatory system is also one that allows the user to interact with it as naturally as possible.

The fourth quality is that the system should work seamlessly in concert with a “privileged” system. That is to say that the AVSA should augment, rather than degrade, service of those with full IIIF technology.

The final quality is that the system should be non-intrusive. We know we will require an interface of some sort to provide the control and make the system as self-explanatory as possible. However, we also know that an intrusive interface will interfere with the visitor’s concentration, reminding them that they are not actually a part of the media space and thus detract from their sense of presence in the media space. The issue here is one of contrast. The interface should offer just the right amount of instruction so that the user has a choice of blocking out the system, paying passive attention to the system or concentrating on using the system.

The next section describes the options that were available for each component of the interface and, based on the qualities, why we chose the options we did.

3.2 Interface

Without a good interface the system will not be usable or even be used. Therefore, we placed special importance on considering our options for this aspect of the AVSA. We broke the interface into three components:

- presentation
- input
- feedback

Presentation refers to informing the visitor of what control they have over the media space. Input concerns how the visitor actually commands the system to perform an action. Feedback refers to how the visitor knows if the requested action was performed and what the results of the action were.

3.2.1 Presentation

The first consideration was how to advise the visitors of what control they have over the media space. There are only two output devices at the videoconference room- the speaker and the monitor. Therefore, the two options for presentation are:

- speech prompts
- visual prompts

Speech Prompts

One approach would have been to provide prompts to the visitor through the audio channel in the form of speech. For example, when the visitor connects to the media space they could be presented with a computer generated audio message like "do action A to connect to Bill, do action B to connect to John, do action C to connect to Tracy.....". This is comparable to the very popular automated attendant style of interface used millions of people everyday. It is a very natural way of receiving information.

Unfortunately, speech output requires sequential processing and is thus time dependent. The problems with using it are threefold. The first being that speech output can be intrusive. Because of its nature, speech requires a significant amount of attention to understand the message. In the case of the AVSA, electronic visitors will be faced with situations wherein they are attempting to listen to members of the media space while also accessing the control information. It is very difficult to block out one stream of speech and pay attention to another. Especially if they are both coming from the same source (i.e. the speaker of the remote node). It is next to impossible to pay partial attention to both.

The second problem with using speech for the output component of the interface is that for every option x, y time is required to listen to the option. Research and experience tell us that long lists presented in this manner frustrate the user.

A third problem is identified as also being related to the time it takes to speak a list of options with long descriptions. Many times during such a list the user is mentally storing the options that most suit their needs. Once the list is has been completed the user searches through their mental store and picks the best of the possible options. This process places an undesirable cognitive load on the user which can be distracting; thus diminishing the experience of visiting the media space.

It is quite clear from our discussion and other research[2][12] that using speech prompts is not the correct route to take for this component. If we were using only the telephone we would have no option and our interface would be compromised. Fortunately, the orphan CODEC has access to audio and video output. Therefore, our options expand to include the visual channel. Our analysis of why speech is not suitable leads us to a suitable option.

Visual Prompts

The underlying cause for rejecting speech prompts is its sequential nature. Visual prompts, on the other hand, are parallel in nature. One can quickly scan a visual image and acquire vast amounts of information. The question is, how would a visual system fare in the context of the AVSA? To answer this question we look at two features that visual displays offer.

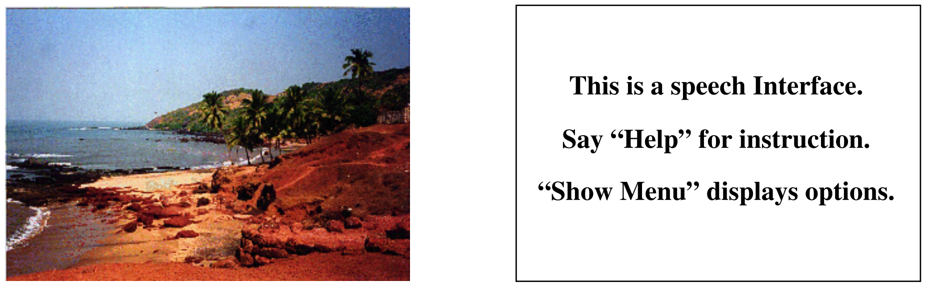
- Options can be presented *instantaneously* minimizing the time required to wait.
- Options can be displayed *persistently* so the cognitive burden of remembering them and then deciding which to chose is reduced.

These features address the second and third problems that would be associated with a speech prompt driven interface. Therefore, the solution will involve providing visual prompts indicating what options are available. These prompts may be in the form of text or some appropriate graphical metaphor. We know, however, that we do not want whatever visual prompt we use to significantly interfere with the other visual tasks that a visitor is involved in.

The discussion provided so far motivates four plausible solutions:

- provide an extra monitor
- provide multiple windows on the same monitor
- provide the ability to switch between dedicated views
- provide the interface overlaid on the media space view

The first option is to provide an extra monitor on which options could be shown (Figure 8). This option does not conform to the LCD quality.



Monitor 1- View of Media Space

Monitor 2- The interface

FIGURE 8 Using two monitors to prompt users does not conform to the lowest common denominator quality.

The second possible solution is to provide a windowing system where the prompts and view from the media space are displayed on one monitor but in different areas of the display (Figure 9). This option eliminates the LCD problem of the first option, but by using this split screen method we would be reducing the screen “real-estate” available to either of the two tasks.

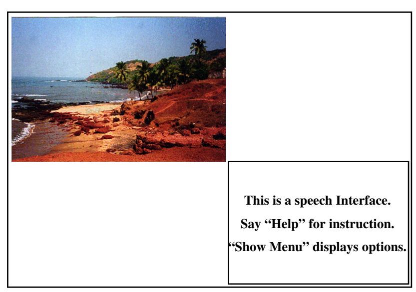


FIGURE 9 Providing multiple windows will yield an unusable interface.

The third possibility is to provide the one monitor associated with the node of the orphan CODEC with the capability of switching what it displays. However, by requiring the visitor to switch video signals between two sources, this third option introduces discontinuities in the visitor’s experience and thus makes the interface intrusive. Even though the audio connection can remain intact and the visual connection is only temporarily broken it is still unacceptable. For example, if a visitor wants an explanation (from a member of the media space) as to what an option will do, they would want to be able to see both the person they are talking to and the interface at the same time. In any case, designers should strive to reduces discontinuities so as to minimize the visibility and thus intrusiveness of an interface as much as possible.

The fourth approach, and the one taken, is to use video overlay technology. This technology allows one to combine computer generated images or video signals with other computer generated images or video signals[16][17]. By using a pre-defined key (chroma or luma keying) parts of any image can be filtered out, much like the overlay used to display information during a televised sports event or the credits/titles on a film.

In terms of the AVSA, this technology can be used to allow a computer generated graphic of options to be ‘overlaid’ on top of the video signal from the media space (Figure 10).

FIGURE 10 Ovelaying the interface reduces time discontinuities and maintains visibility

The important parts of the graphic (i.e. the parts that indicate the options) will be shown to the visitor while the non-valuable sections can be filtered out. By using this technology we are exploiting a third feature of video:

- Video is spatially dependent so when presented with multiple images it is possible for a user to ignore or partially concentrate without much effort.

Take an X-Windows interface for example. If two windows A and B are present and one wants to concentrate on window A, one simply has to turn one’s head or shift one’s eyes. Compare this to a speech prompted interface. There is no equivalent action. You cannot close one ear or shut out audio from any one stream. It is possible to concentrate more one audio stream to reduce the impact of the other especially if they are coming from different locations, but requiring a visitor to do this for long periods of time would be irritating, stressing and tiring.

3.2.2 Input

The second aspect that needed to be considered in designing the interface was providing some mechanism by which the visitor can indicate which option they wished to choose. For the interface to be non-intrusive, and thus usable, it was clear that we had to use an accurate form of input. Unfortunately, the most accurate ways to provide input (i.e. a keyboard press, a touch screen, pointing device, etc.) would require extra equipment to be placed at the videoconferencing site. This does not conform to the limited equipment design constraint. Therefore, these forms of input are not practical options.

The only input receptors available at traditional videoconference rooms are:

- CODEC touch-tone keypads
- camera
- microphone

Touch-tone keypads

With today's technology, touch-tone keypads would offer the most accurate interaction when compared to the two other receptors. However, CODECs do not necessarily have keypads and they certainly do not have the standardized touch tones of telephony. Consequently, by using this input device we would not be adhering to the LCD quality. In addition, it is not a very natural way of interacting and thus would be intrusive.

Camera

The second receptor, the camera, would require the AVSA to be equipped with a gesture recognition algorithm. For our purposes, accurate use of gesture recognition would impose a steep learning curve on the visitor. This would not conform to the self-explanatory quality. The accuracy problem would be compounded because accuracy of gesture recognition algorithms is highly influenced by the environment.

Another problem with using the camera stems from the fact that there is only one camera at the videoconferencing room. Members of the media space will see whatever the AVSA sees- namely, the gestures. The gestures seen by the members of the media space will, most likely, seem irrelevant and thus be distracting. Depending on culture, they may even be inappropriate. One way around this problem would be to temporarily discontinue the video seen by the media space member. This would solve the original problem, but introduce disturbing discontinuities for the media space members. Both problems would violate our seamless interaction quality.

A final problem with using the camera is that since the camera's field of view is limited, the user would be restricted to a certain area within the videoconference room. This would violate the ubiquitous quality.

Microphone

The third input receptor, and consequently the one chosen, available at a traditional videoconference room is the microphone. Since most people have the ability to talk, using speech would allows adherence to the LCD quality. It is ubiquitous because the input device travels with the person and the input receptor is always present at an orphan CODEC. It can be made non-intrusive by placing an omnidirectional microphone at the orphan CODEC. Finally, since speech is a very common way to communicate, the interface can be made self-explanatory. One potential problem is that the audio commands may disturb members of the media space.

Ideally, we wanted to equip the AVSA with a speaker-independent, continuous-speech automatic speech recognizer (ASR). The reality, however, is that today's ASR technology is not very accurate. Especially in our context where the receptor is an omnidirectional microphone and the noise levels of the environment are uncontrollable. However, despite this problem and based on our evaluation of the possible input mechanisms speech is still the best option.

We decided that we could sacrifice the flexibility of the ASR in order to increase the accuracy. As a result, we decided to use a small vocabulary discrete-utterance recognizer.

3.2.3 Feedback

The final consideration in designing the interface is indicating to the visitor what their input accomplished. For the same reasons outlined in Section 3.2.1, it was clear that feedback should be provided in the same form as is provided by the presentation of the system- as a visual stimulus.

3.3 Architecture

Having defined the basic look and feel of the interface, the next task is to define the basic look of the system architecturally. The two details we consider are:

- where the AVSA fits in to the videoconference room to media space communication model
- the internal structure of the AVSA

3.3.1 Location of AVSA

It was quite clear from the design constraint that the AVSA must be located at the media space. Figure 11 shows how the AVSA should be connected to the media space.

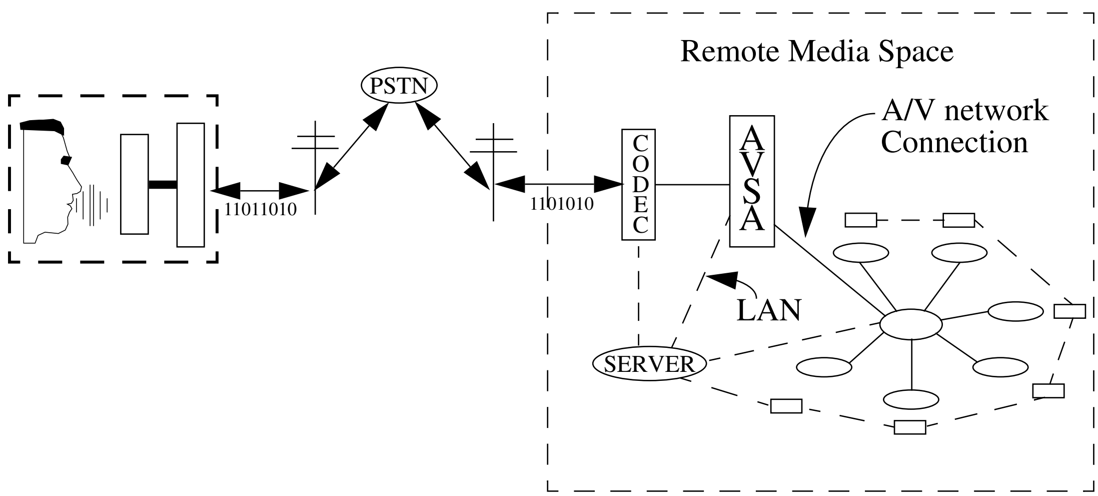


FIGURE 11 The AVSA in relation to the media space and the videoconference room.

3.3.2 The AVSA structure

The AVSA itself is divided into three layers (Figure 12):

- the interface layer
- the communication layer
- the translator layer

The interface layer

The interface layer allows a person to pass and receive information to and from the AVSA. Section 3.2 provides a basic description of how it will work.

The communication layer

The communication layer allows the AVSA to pass and receive information to and from the media space. The AVSA will use the Ethernet communication channels already used by the media space TP client to communicate control information. The AVSA will also be connected to the A/V network so that the AVSA can send the visitor the composite image containing the interface overlaid on the view of the media space.

The translator layer

The translator layer is the link between the interface and communication layers. In terms of the AVSA the two jobs of the translator are:

1. based on what is recognized by the speech recognition system decide what information needs to be sent to the media space.
2. based on what the media space sends to the AVSA decide what is to be displayed.

The translator basically parses data coming from the IIIF server and stores this data into appropriate structures. It also constructs commands to be sent to the IIIF server based on a users requests.

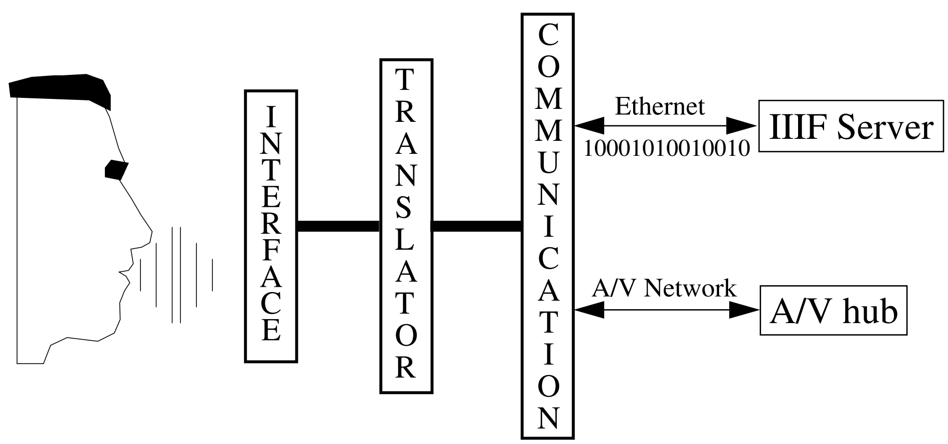


FIGURE 12 The different layers of the AVSA

3.4 Summary

In this chapter we introduced the five qualities with which we want to design the system.

- it should cater to lowest common denominator
- it should be as ubiquitous as possible
- it should be self-explanatory
- it should work seamlessly in concert with the “privileged” system
- it should be non-intrusive

We then outlined the basic look and feel of the system based on these qualities. The AVSA will receive input as speech commands from a person at the videoconferencing room. It will prompt the user and provide feedback through the video channel by ovelaying the text on the media space image.

We identified the location of the AVSA in relation to the media space and videoconferencing room.

Finally, the basic architecture of the AVSA is laid out in terms of three layers- the interface, communication, and the translator.

CHAPTER 4 Stage 2: Rapid Prototyping

This chapter describes the second stage in the development of the AVSA - rapid prototyping. Our goal here was to develop a system as quickly as possible to prove our concept and so that we could obtain user feedback that could be used in the development of subsequent iterations and more complete versions of the system.

4.1 Ingredients

The first task was to gather, based on the choices made in chapter 3, all the technology which we needed to build the system- the ingredients. The ingredients were subdivided into three main categories- the hardware, software and last, but not least, developers. In accordance with the goal of this stage we first looked at what was immediately available to us and then what could easily be acquired.

4.1.1 Hardware

We identified the following hardware requirements.:

- a video overlay board that accepts an external NTSC video signal and an internal computer generated signal. The composite of these images must be accessible externally as an NTSC signal.
- a sound processing board that accepts an external analog audio signal.

- a network interface card.
- a computer to combine all of the components and on which the system is to reside.
- access port to the local area network (LAN).
- appropriate audio and video connections between the visitor and the media space.

4.1.2 Software

The following software requirements were identified:

- video overlay software and application programming interface (API).
- automatic speech recognition (ASR) software and API.
- network interface software.
- software developer toolkits on which to build the interface, the translator and communication utilities and in which to integrate the video overlay, speech recognition and network technologies.

4.1.3 Developers and reviewers

As previously mentioned, the interface is the most important component of the system. Careful consideration has already been taken in the choosing the technologies to be used. To ensure that our system is developed with the user in mind we decided that the system should be developed by people with as diverse a background as possible. We should have people who have never used this type of technology before, people who have used this type of technology, and people from different fields and levels of education. We also realized that it is not always possible to cover all the basis, therefore we decided that after successive iterations users should be brought in to test the system and/or review our design and system.

4.2 Recipe

The process of building the system is as follows (testing all hardware and software as required):

- choose, gather, install the appropriate ingredients.
- connect the AVSA to the media space through the audio and video links.
- connect the AVSA to the Ethernet.
- build TCP/IP communication subsystem to communicate control information via the Ethernet.

- specify communication protocol for communication between the AVSA and the media space. Alter the media space so that it conforms to this new protocol.
- develop the translator based on the communication protocol.
- integrate the ASR.
- develop the interface.

4.3 Preparation

The following sections discuss the actual preparation of the prototype AVSA based on the recipe laid out in Section 4.2.

4.3.1 The hardware and software

After considering all the options available to us we decided to develop the prototype on the personal computer platform. We acquired an IBM PC clone running Microsoft Windows 3.1 to be the dedicated server.

We obtained and installed a video overlay board produced by Roctec Electronics Limited called RocGen VGA. The overlay hardware came with a device driver and API to allow us to integrate it into the system.

For the sound aspect of the system we decided to use a Sound Blaster 16 card to process the audio signal and a speech recognition software called Voice Assist that was developed by Creative Technology Limited.

To connect to the Ethernet (our LAN) we obtained a network interface card by 3Com Corporation as well as their device drivers. A shareware PC-NFS (network file system) software to allow the PC to access the UNIX system files that were crucial to the system was also acquired. The Windows Sockets API (WinSock API) was also obtained to allow us to build communication utilities for the windows application.

To build and integrate all the components we planned to use Microsoft's Visual C++ (MSVC) programming environment and Windows Software Development Kit (SDK).

4.3.2 The development team

The development team we gathered consisted of:

1. William Buxton - A musician and expert in the field of user interface design. Professor Buxton introduced the idea of the AVSA and was to play an active role in reviewing the system and, as a user of the former system, coordinate requirements with other users.
2. Koichiro Tanikoshi - A visiting scientist from Hitachi Research Laboratory in Japan. Koichiro brought many years of experience in the field of human-computer interaction (HCI) to the team. His role in the project was to initiate development on the hyperdoorway (the system which was eventually developed into the AVSA).
3. Anuj Gujar - An HCI graduate student whose past experiences included a specialization in software systems. His role was to take an active part in the development of the hyperdoorway and to supervise and develop the AVSA.
4. Shahir Daya - An electrical engineering undergraduate student. His role was to aid in the development of the utilities that allowed communication to the media space and the eventual interface for the AVSA.
5. Jeremy Cooperstock - An HCI graduate student with the departments of Electrical Engineering and Computer Science. Involved in the initial start-up of the AVSA Jeremy's main role was to help customize specialized media space services for the AVSA.
6. Tracy Narine - Administrator for the iiif system. His role was to help integrate the AVSA into the local media space.
7. Radek Nowicki - An undergraduate student with the department of Computer Science. His role was to aid in the initial stages of developing the interface.
8. Don-Jun Wang - An undergraduate student with the electrical engineering department. His role was to help in the initial stages of developing the communication utilities.

From this list it is clear that we accomplished our requirement of obtaining developers from various backgrounds. We had people new to the technology, people familiar with the technology and people of varying levels of education. We were confident that by using this team we could obtain user feedback while developing the system thus minimizing the impact of the user feedback which will be obtained when the prototype is completed.

4.3.3 Linking to the Media space

The first task to be done was to register the AVSA with the media space. By registering, we were updating a database to tell the media space that there is a device named *av-server* and which four wires (2 audio and 2 video) will hook the AVSA into the media space. Once registered, the appropriate wires had to be run to the PC for hook up.

A default connection to the AVSA was also set up. So that when a visitor connects to our CODEC they will see whatever interface the AVSA wants them to see. In the initial stages this display was a view from a camera pointing at an appropriate sign. Scripts were also written to allow the AVSA to tell the media space when and how to connect an electronic visitor to a room within the media space through the AVSA.

The second task was to install the Ethernet card and install the drivers. Then the LAN had to be informed of the new machine. The NFS software was also installed to allow UNIX directories containing crucial media space information to be mounted on the PC.

4.3.4 Sending requests to the media space

The next step was to build communication utilities to allow the AVSA to send and receive control information to/from the media space from/to the PC. These utilities had to be compatible with the communication protocol of the media space communication utilities. We use the WinSock API network programming interface to develop these utilities. It is based on the “socket” paradigm popularized by the Berkeley Software Distribution of UNIX.

These utilities, which came to be known as the hyperdoorway system, allowed the AVSA to send requests to background processes of the media space and receive responses from the background processes based on these requests.

4.3.5 Querying the media space

The replies that were being received by the hyperdoorway system were not formatted in any uniform way. In actual fact the replies were confirmations that the request had been processed. To the hyperdoorway they were meaningless and useless pieces of information. The system we wanted had to be able to process the replies into meaningful pieces of information so that further request could be made of the media space. It was at this point that the AVSA was born. By negotiating a protocol with a developer of the other media space components, Jeremy Cooperstock, we came up with a simple protocol by which the AVSA should communicate.

How does it work?

The media space consists of two main systems. The first was the IIIF and the second was the reactive room[7]. The IIIF is a software/hardware system enabling one to make connections between members of the media space. The reactive room consisted of a hardware/software sub-system that allowed one to control A/V resources within a room. The reactive room subsystem has the ability to communicate with the IIIF. Since we were trying to complete this stage quickly, we decided that we should speed up the prototype process by dealing with just the reactive room and communicate with the IIIF through it (Figure 13). A translator was then built into the AVSA to parse information, based on the established protocol, into appropriate data structures.

The basic communication is as follows:

1. The AVSA talks to the IIIF via a reactive room daemon called the telepresence (TP) daemon (*tp_daemon*)
2. The AVSA talks to the reactive room through a system of daemons, that are a part of the reactive room subsystem, to control the reactive room's resources.

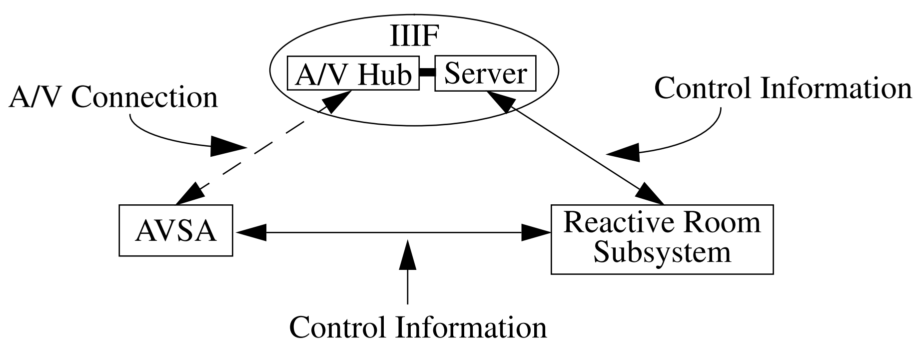


FIGURE 13 The AVSA talks to the IIIF through the reactive room subsystem.

4.3.6 Integrating the ASR

The ASR we were using did not have an API. Once the ASR was initiated the application which was *active* was the one acted upon. For our purposes this application is always the AVSA. All audio will go through the ASR. When the ASR recognizes a word, it maps that word into a corresponding keyboard press. The translator was coded to recognize this keyboard press, access the appropriate information to send to the media space and then, through the communication component, send the information.

4.3.7 The first interface

Since the interface is a layer independent of the communication layer, the first AVSA interface was developed as part of the hyperdooway system in parallel to the communication component. It presented a hard coded list of captions in a box at the bottom right section of the screen as a textual list descriptions preceded by a number (Figure 14). The idea was that the visitor would say the number. If the ASR recognized a number, the command associated with that keypress was sent to the media space. As the translator and communication protocol of the AVSA system were developed a menu hierarchy (Figure 15) was formed through which a visitor could traverse and access services. The services were no longer hard coded as the media space was queried for services at each level.

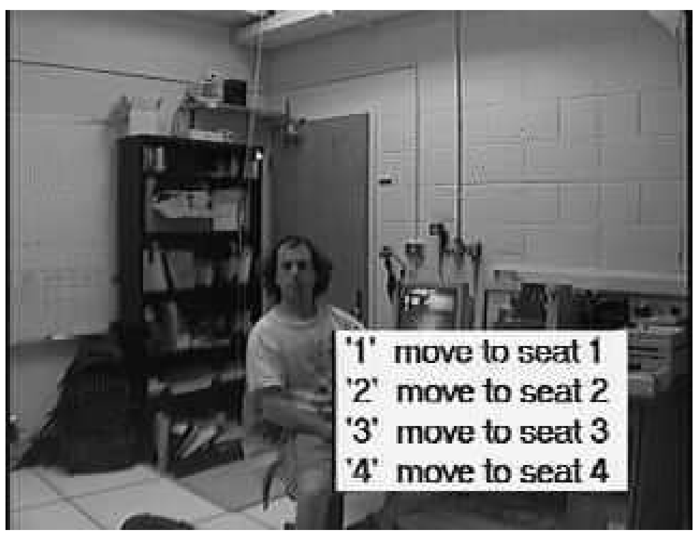


FIGURE 14 This view of the first interface shows the options available to a visitor who has chosen a reactive room service allowing them to move to a different seat.

The hierarchy of services

When a visitor calls a media space's CODEC, the AVSA presents the visitor with the option of calling a member of the media space. If the visitor says they wish to call someone, the AVSA asks the IIIF who they may connect to and then presents these options to the visitor. When the visitor chooses to connect to a member, the AVSA instructs the IIIF to make an A/V connection between the visitor and the member. At this point an option is presented to allow the visitor to disconnect from the member's space. If the member happens to be the reactive room, the reactive room daemons are queried for services. If services exist they are presented. Some services will have further options which are displayed when and if the service is chosen.

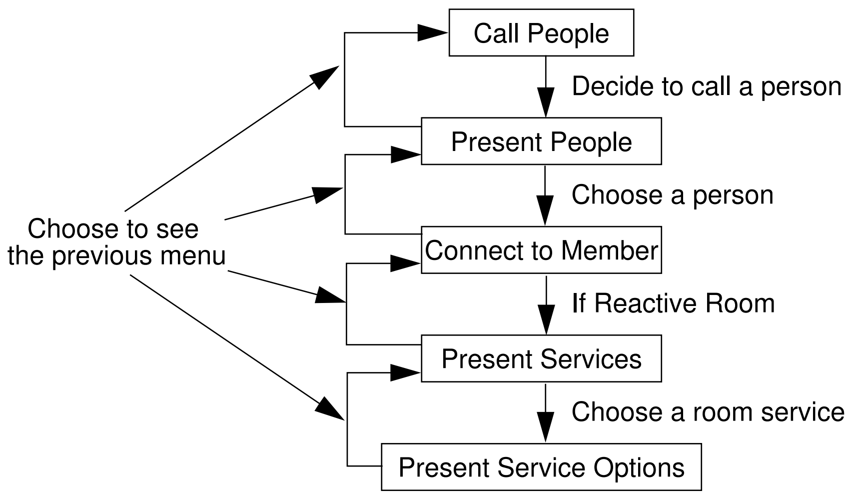


FIGURE 15 A schematic of the menu hierarchy

At this point the basic system had been developed and we decided to take a step back and look at how we could improve the actual interface.

4.4 Enhancements

In improving the interface we went through several iterations. At the end of each iteration we obtained input from members of the IRG and/or the development team. This input helped us make decisions on how to improve the system for successive iterations. The evaluation was done informally by exposing the system to the user group and noting their comments. Approximately twelve people were involved in the evaluation leading to the enhancements discussed in this section.

We explored two different ways of improving the interface. The first was to use a more descriptive interface making it more self-explanatory. The second was to improve the textual interface to make it more effective. The following sections describe the enhancements and iterations which we implemented.

4.4.1 A more descriptive interface

With the text interface we observed that it was not always apparent to the visitor what would happen if they chose an option. For example, in the reactive room if the visitor chose the service *seat changing* they were presented with the textual list of different virtual seats to which they could move allowing them the ability to get a view of the room more appropriate to their social role[13]. This list did not tell the visitor anything about what they would see when they moved to that seat. The visitor would have to go through a trial and error process to get acquainted with the room.

First iteration

Our first impression was to provide a more descriptive caption for the options. So instead of the captions “Move to seat one, Move to seat 2, Move to seat 3......”, the visitor would see “View the blackboard, View the door, View the Mac....”. This certainly was more descriptive. However it was also very limiting. What if two things could be seen from a particular seat? Would the caption have to include both objects? What if two seats provided a view of the same object? How would we provide a description that allowed the visitor to choose the best one for their purposes? How could the description be updated to reflect a change in the position of the seat? Some cameras in the media space are motorized to allow them to pan through the room. How would the description be updated to reflect these changes? We also had to keep the descriptions as short as possible so that the interface did not become too intrusive.

We decided that a textual list was not the way to add description. Instead graphical symbols may be more appropriate.

Second iteration

Our first attempt at adding a more graphical interface was to generate an image file containing a map of the room with its objects labelled. This image was displayed when the visitor chose to access the seat changing service (Figure 16). Now the visitor could visualize what they would see if they moved to a particular seat. The need for lengthy descriptions was eliminated. Unfortunately, there were still problems with this interface. How would the image map be updated to reflect changes in seat position? How does the visitor know which seat will give them the best view of non-static objects such as people.

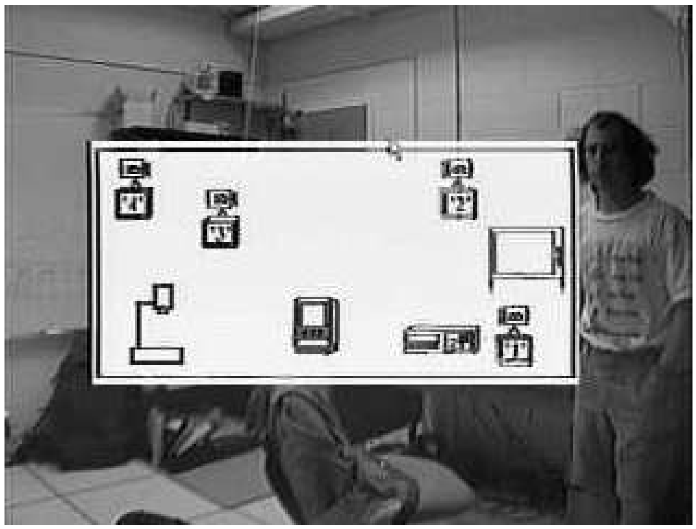


FIGURE 16 Using the floor plan to facilitate orientation.

Third Iteration

The third iteration added snapshots to the interface, updated every *n* seconds, containing views from each seat (Figure 17). This solved most of the problems of the first two iterations. Lengthy descriptions were no longer required and the visitor got up to date views from each camera.

Unfortunately, much of the view of the media space is obstructed by the interface making it intrusive. We could not decrease the size of the images either because, as mentioned before, the CODEC degrades the quality images. Smaller images would be useless for their intended purpose to the visitor. We also observe that as the number of options increase more and more space would be used by the interface making it increasingly intrusive.

Another potential problem was that by eliminating the textual description we were risking the possibility that a user may not understand the purpose of the interface.

FIGURE 17 Snapshots (the four smaller images at the top) provided enough information to eliminate trial and error and lengthy descriptions, but the interface is now too intrusive.

In addition, one of our goals in developing the AVSA was to make it as flexible as possible so that it could be integrated into any environment. The type of customizations we have described require a static environment. These customizations can be worked out for a particular media space when and if required.

Based on the above problems, we decided that improving the interface through a more graphical approach was not the correct path to follow. Therefore, we turned our focus to developing a more generalized system through a text interface.

4.4.2 Improving the text interface

Having decided that the generalized text interface was the style of interface that we should improve, we looked at the deficiencies of the original interface described in Section 4.3.7

Feedback

The first deficiency was that of feedback. Our initial thought was that since all of the actions resulted in some visual and/or audio shift we would not require any extra feedback. This, however, was not the case. Users wanted to see what word was recognized (incorrectly or correctly) by the ASR. Users wanted some indication as to what action was being performed. Users wanted to know if the action had failed.

To provide this information we decided to implement the *banner*. The banner was a one line description of text displayed at the top of the screen (Figure 18). It told the user what action was being performed and if the action was successful or not. The ASR we were dealing with did not have an API so we could not provide this information to the AVSA to be displayed in the banner. Instead we overlaid the ASR's GUI on top of the video image. This GUI showed activities of the ASR sufficiently enough to provide the feedback which we required.

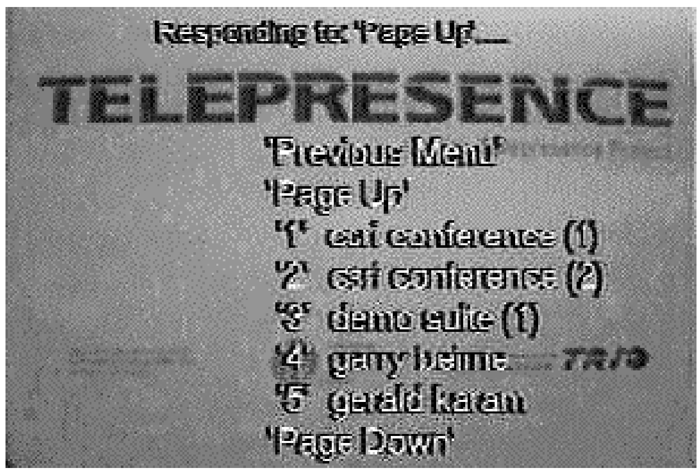


FIGURE 18 The banner, at the top if the screen, provides feedback indicating what was chosen.

How do I direct my speech?

One of the difficulties with using speech as an input mechanism and the medium through which to communicate with members of the media space is the issue of how the AVSA knows when the speech is directed at it and when it is not directed at it. When we first designed the interface we decided that after a certain time-out period (about 10 seconds after the last command seemed to work well) the list of options would disappear. Any time the options were not visible, the AVSA would not perform any actions. The only way to activate the AVSA again was to bring the menu up again by uttering the words “show menu”. This mechanism made it easy for the system to recognize if the visitor was talking to the AVSA or to someone in the media space. It is similar to the protocol used in the popular television show Star Trek (i.e. instead of “computer”).

There were some problems with this mechanism for our case. Unlike Star Trek’s computer, today’s technology lends the AVSA limited ability to understand the user. Therefore we cannot realistically design the system so that the user can converse with the AVSA as if talking to a real person. Relatively few commands are available. This is the reason we display available options through visual prompts. By fixing a time before which the menu will not disappear we are introducing a situation in which it is possible that the AVSA will not know who the visitor is talking to. For example, if the visitor connects to the media space, they do not want to wait fifteen, ten or even five seconds before they start their conversation. One solution would be to simply make the menu disappear as soon the AVSA completes an action. The problem with this solution is that there are several cases in which the visitor will be travelling two or three levels into the menu structure before wanting to converse with a member of the media space. It would not be appropriate to require the visitor to say “show menu” at each level. Our solution is to provide a “hide menu” command that will allow the visitor to remove the menu from the display indicating to the AVSA that the visitor is now speaking to members of the media space instead of the AVSA.

Option Scrolling

There were a number of cases in which the number options that were available was very large. For example, the video cassette recorder (VCR) had a list of fifteen possible control options. In this case displaying all the options on the screen would occupy the whole screen and make the interface too intrusive. As a result, a scrolling system was implemented. We set a limit of five control options at any time plus *page up*, *page down* and *previous menu* (Figure 19). Page up would display the next five options. Page down would display the previous five options. If there were five or more options available, five options were displayed at all times. For example, if there were eight options. Options 1-5 would be initially displayed. When visitors indicated that they wished to scroll down options 4-8 would be displayed not just 6-8. This ensured that we maximized the number of options that the visitor could access at any one time.

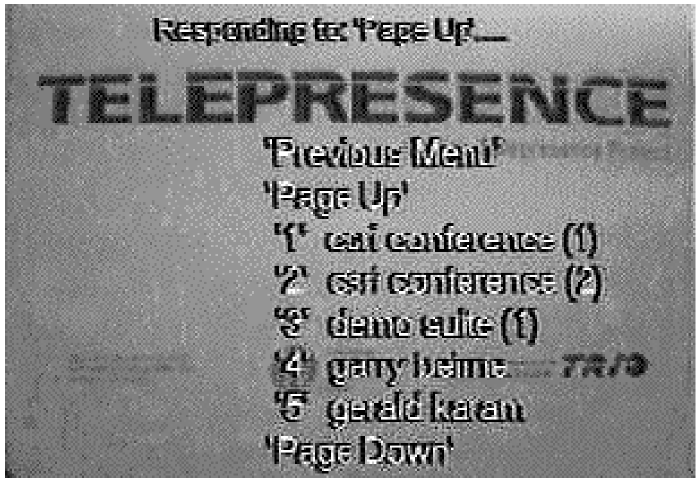


FIGURE 19 Page up and page down commands are used to scroll through the menu.

The background box

Another deficiency was that of the general appearance of the visual prompts. First, all the options were displayed in a solid box. Also, the text of the feedback banner was contained in a solid box. We felt that this emphasized the presence of the list too much. We wished to make it more subtle. As a result, we decided to eliminate the box and just display the text captions with the associated command to be uttered.

The text

By removing the background box we had created another problem. The text, which was displayed as solid black, was invisible or barely visible in some situations. For example when the option list happened to be overlaid on top of a black desk, or when the banner was displayed in a very dark area of the view (usually due to lack of light and/or a dark colored ceiling).

We decided that the best way to deal with these problems was to adopt a technique used in the television/movie industry. The text would be displayed as two colors. One color outlining the other. We experimented with various colors. Some colors did not offer enough contrast. Some colors *bled* when passed through the CODEC. Some were simply not appealing. The combination that worked the best was white text outlined with black.

Our conclusion regarding the colors was supported by the television/movie industries who frequently use this combination of colors for close captioning.

Static and dynamic options

Options were presented as a number followed by the caption describing the result of choosing that option. As described earlier this was done to maximize the accuracy of the ASR. Some users remarked that by restricting the user in this manner we were actually diminishing the relevance of using speech to make the interface natural. We took a closer look at the interface. As a result, we realized that the options *previous menu*, *page down* and *page up* were actually static options. That is, they either appeared or did not appear on each menu. Therefore we should still be able to reach high levels of accuracy by requiring the visitor to say the actual action instead of the number associated with that action. As a result, these three static actions were invoked by uttering their caption instead of a number that corresponded to their action.

The Commands

Another problem with the interface was in the way the commands relating to the options were presented. It was not clear what part of the list was to be uttered to invoke the command. This problem was compounded by the addition of static and dynamic options.

We decided that a simple self-explanatory indicator would rectify this situation. As a result we borrowed a convention used in literature to signify speech. We decided to represent the utterable words enclosed in quotes. This proved to be a very effective way of representing the utterable commands.

4.5 User impressions

We were now at a point where the AVSA had been developed sufficiently enough to show it to a larger group of people in order to obtain their comments on:

- how useful they think the system is
- what could be changed or added to further improve the system

Although it was not plausible to perform any formal user evaluation, we were able to learn a lot from the comments of people who informally reviewed and tested the system. Thirteen people reviewed the design of the system and were exposed to the system. Nine of these people rated their expertise on a scale of 1 to 5 (5 meaning an expert in the area) in this area of work. Three rated themselves as 5, four rated themselves as 4 and two rated themselves as 3 on the scale. The other four reviewers did not rate themselves, but said were moderately familiar with videoconferencing systems. Another set of ten people from the IRG, the development team and the DGP lab observed and/or used the system in an informal setting to provide even more information regarding the functionality and usability of the system.

4.5.1 The concept

The immediate reaction of most people who saw the system was very positive. They were very interested in the concept and its contribution to media space accessibility. In addition, from our own experiences in using the system and comparing it to the previous situation it was quite obvious that we were on the correct path.

4.5.2 The system

The system itself was also quite well received by the people we questioned. One regular user commented that if the ASR was more accurate they would prefer this interface to the TP application.

Despite the obvious optimism, they did suggest some changes to the interface to improve their ability to navigate through the menu structure. Our experience and observations of them using the system also gave us ideas on how to improve the system.

One of the major shortfalls of the system was the ASR. The ASR which we had obtained was a beta version which was given to us. As such its performance was not as good as an ASR should be for our type of system. We required a very high rate of accuracy (95% or more). We were only achieving approximately 80% accuracy with a speaker-dependent ASR. It was clear that if our system was to be deployed into the field we would have to carefully choose an appropriate ASR.

4.6 Summary

This chapter described the development of a prototype AVSA from gathering the ingredients of the system, planning how to put it together, actually building the system, revising the system based on evaluators' comments and then obtaining further comments to be integrated during the next stage of development of the AVSA. Based on these comments we learned that users accept the concept of the system and the system itself. There were two major criticisms of the system. First was that the ASR was not performing at a level that was acceptable for the needs of our system. Second, the PC system was very unreliable.

CHAPTER 5

Stage 3: Self-Contained System

The second stage in the AVSA development allowed us to produce a usable interface that was readily accepted by the various people who reviewed it. The goal of this stage was to take user input from the previous stage and integrate it into the system and more importantly wrap the AVSA up into a nice, stable and self-sufficient package.

The AVSA of stage two relied heavily on the state of the reactive room subsystem. If the reactive room malfunctioned the AVSA did not work. In addition, the communication between the PC/UNIX environments made the AVSA unstable in some cases. It placed heavy requirements on an environment not meant for such tasks. Finally, since the PC did not support multiprocessing it was not possible to set up a monitoring system (i.e. daemon) for the AVSA to accept requests from the media space. This chapter describes the third stage in the development of the AVSA- developing an integrated self-contained system on the UNIX platform.

5.1 Ingredients

The basic parts required for this stage are the same as those required in stage two and are described in Section 4.1. However since we wanted a more stable and self-sufficient system, the actual ingredients acquired would be different in that they will all operate in the UNIX environment.

5.2 Recipe

The recipe was slightly different from stage two's. Fortunately, even though the ingredients would be different we would still be able to port some components, like the translator, since they will remain constant. The recipe for this stage is as follows:

- choose, gather, install the ingredients.
- connect the new architecture to the media space through A/V links and Ethernet.
- investigate and use established TCP/IP utilities to communicate control information via the communication protocol established in stage 2.
- port the translator to the new system.
- integrate the ASR.
- port the interface, integrating suggestions from users during the previous stage.

5.3 Preparation

Once again we will use the recipe to build the new self-contained system. The following sections describe our efforts.

5.3.1 Ingredients

Since one of the aspects we wanted to improve in this stage was the communication, we decided that we should implement the AVSA in some multiprocessing environment. We also wanted to make it more compatible with the environment in which the media space was already constructed. This would help create a more stable communication system. Our choices were thus limited to the UNIX platform. Since we had convenient access to Silicon Graphics Incorporated (SGI) hardware and software we decided that development should be done in the SGI environment with the IRIX operating system. As a result, we acquired an Indy workstation on which the AVSA would reside.

We also needed video overlay technology for the Indy that would allow the AVSA to perform the video overlaying which we required (described in Section 3.2.1). For this we acquired an Indy Video card and its associated video library.

For the speech recognition we acquired an ASR developed by SGI. Indy's come with a standard CD quality sound processing card with an external input so we did not have to purchase any extra hardware or do any installations to process the audio.

Indy's also come with their own standard Ethernet card. Hence, no special hardware or software installation for network connectivity was required.

To build the interface we decided to use the tcl/tk development environment. We would use the standard C compiler to combine all of the different technologies. By using these standard software development environments we were assuring ourselves that if, in the future, the system was to be ported to another platform, the interface would not have to be redeveloped from scratch.

5.3.2 The development team

The development team was downsized at this point. We did not required as much input during the development into the interface. Instead we needed an efficient team that could develop the system quickly. The team consisted of:

1. William Buxton - his role stayed the same as that in the second stage.
2. Anuj Gujar - his role was to acquire all the ingredients and build the AVSA on the SGI platform
3. Akil Nasser - an undergraduate student with the computer science department. His role was to assist in porting the interface and in working with the new overlay technology.
4. David Audrain - An undergraduate student and exceptional C programmer from France. His role was to help develop a driver for the laserdisc player for integration into the video on demand subsystem.

5.3.3 Linking to the Media space

We used the same IIIF wires used in the first prototype to connect the Indy to the media space. We also used the same Ethernet drops. Since the Indy runs on the UNIX platform, no special software was required to enable file access as in the first prototype. This proved to be an important point as it made the system much more stable, reliable and flexible.

5.3.4 Sending requests to the media space

For the prototype system we had to build communication utilities to allow the AVSA to communicate with the various media space systems. The reason for this was that there were no established utilities for the PC platform. Since the new system will now reside on the UNIX platform, and the main core of the media space is developed on the UNIX platform, we can use the established UNIX communication utilities for communication.

By using the established communication utilities we can separate the systems so that the AVSA will not be dependant on the reactive room subsystem(Figure 20). The former situation was acceptable for the prototyping stage, but the new setup is more logical. In this new configuration, the AVSA communicates control information to the IIIF system directly instead of through the reactive room subsystem as in the last stage.

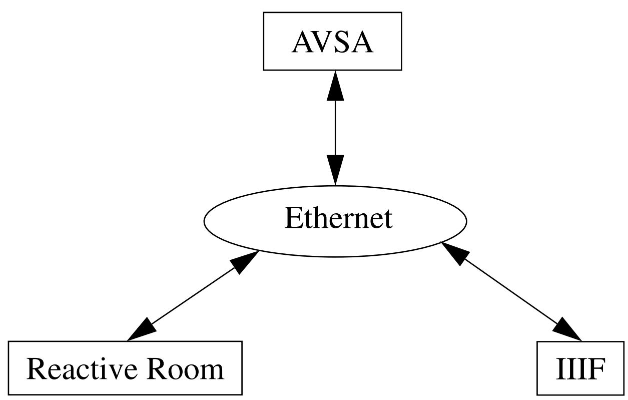


FIGURE 20 The AVSA now uses the Ethernet to communicate with the IIIF directly instead of through the reactive room.

5.3.5 Two-way communication

The Indy has much more processing power than the PC and the UNIX platform has many more processing capabilities than Windows. This should simplify the communication and make it much more stable.

In the last stage, the AVSA could send requests to the media space and receive responses based on the these requests. To enable this on the Indy the translator of the second stage had to be ported to the Indy platform.

In addition, there were some cases in which we wanted the media space to send requests to the AVSA without being requested to in the first place. For this reason we decided convert the AVSA communication layer into a daemon. In this way, after some simple modifications, other media space systems would be able to communicate with the AVSA at will.

5.3.6 The ASR

The process of integrating the ASR would be the same as the process in stage two because there was no API. However, since we were using a Silicon Graphics Incorporated (SGI) ASR package, and we had close ties with SGI, we were planning on working closely with the developers of the product to develop an API and tune the ASR to our needs. Unfortunately, the project under which the ASR was being developed at SGI ran into problems and was consequently terminated.

We did search for other ASR options. Vendors were contacted and the options were evaluated. The reason that we decided not to purchase a commercial ASR is that when we took into consideration:

- the cost of the technology we required (which averaged approximately $4000 U.S.),
- the time it would take to integrate the software into the AVSA and,
- the impact on this research,

we felt that it was not sufficiently beneficial.

5.3.7 The interface

One major task of this stage was to port the interface from the PC to the Indy. We first salvaged as much of the logistics as possible from the Windows version. This basically meant that the translator and the logic behind the visual prompts were preserved. The actual visual prompts had to be reimplemented in tk/tcl. A new video overlay system also had to be integrated into the system using the new overlay hardware and software.

5.4 Enhancements

At the end of stage two, the system was evaluated by several people (as described in Section 4.5). From their comments we decided to make various enhancements to the AVSA. The enhancements can be divided into two categories:

- improving the interface
- adding functionality

The following sections present a detailed discussion of these enhancements.

5.4.1 Improving the interface

Based on the evaluation of the AVSA at the end of the second stage, the evaluators felt that several aspects of the interface could be modified to improve the look and feel of the interface. The following sections detail these enhancements.

Translucent background

In the last stage we removed the background box to make the options list less intrusive. Although our evaluators agreed it was less intrusive than the previous iteration, we observed that when there were few options people using the system did not always realize that options were showing. We observed this situation when there were few options available to the user. For example, when a visitor connected to a room with no services, the only option was to disconnect from the room and go to the last menu. In this case the interface was not sufficiently visible. In essence the interface was becoming a part of the media space, so much so that the visitor would not know their options.

The video overlay technology we were using gave us control over opaqueness of images. We used this to our advantage and solved the problem by presenting the list of options in a translucent box (Figure 21). This way there was always a more visible hint as to what was part of the interface and was part of the media space view.

Commands and captions

Even though quotes helped users distinguish between the words to be uttered and the captions, users still felt that an additional feature should be added to make the distinction clearer. As a result, we decided that the words that are to be uttered will, in addition to the quotes, be yellow with a black outline and the captions will be white with a black outline.

In this way the function of the quotes is to signify what is the caption and what is to be uttered, while the color difference serves to make the distinction between the two clearer.

FIGURE 21 The translucent background helps identify the option list.

Location of options list

In the last stage the list of options was located on the bottom right hand corner of the visitor's monitor. This location presented a problem which will now be discussed.

The area occupied by the list was variable. The height depended on the number of options that were available, while the width depended on the string length of the longest caption. It was this variable nature which caused problems for users. Users wanted to be able to scan the list quickly, choose an option and find the associated word to utter. The constant horizontal and vertical shifting of the list, and thus search for the beginning of the caption as well as the first caption, seemed to slow users down as well as irritate them when they were going through several levels of the menu hierarchy.

From this analysis we concluded that the top left corner of the screen might be the best location for the list of options. This way the users would always know where to start scanning the list. This however was also found to be unacceptable to users. They found it too intrusive because they also scanned the view of the media space from left to right, top to bottom.

It was quite obvious from users comments that the best solution was to place the list in the bottom left corner of the screen. This eliminated the horizontal shifting. Vertical shifting still occurred, but based on our experience of using the interface and the comments of people we asked to evaluate the system, by exposing them to the four possibilities, it was found not to be as troublesome.

Feedback

All of the people asked to review and evaluate the system liked the idea of using a banner to provide feedback. However, they also felt it would be more effective to indicate what option is chosen by somehow highlighting the option in the list instead of presenting a message like “Responding to: 1. Call people” in the banner. As a result, we implemented a flash mechanism. When the user picked an option the color of the inner part of the outlined text would change periodically. Alternating between the color of the outline and the original color (Figure 22).

This feedback is more direct. It also reduces the number of types of feedback that are presented through the banner, thus reducing the cognitive load on the user. In short, the feedback was now presented in a much more effective way.

FIGURE 22 We accomplished flashing by periodically switching between the outlined text, shown on the left, and solid text, shown on the right.

Speeding up navigation

When a visitor visits the reactive room the AVSA queries each daemon for available services. Depending on the traffic over the LAN and the activity of the reactive room daemons, this querying process sometimes took a long time. This left users in a state of confusion wondering if the system had, for example, crashed. To rectify this situation we decided to implement a caching mechanism. With this caching mechanism if the AVSA detected a change in the reactive room it would reload all of the services that belong to that room into data structures. If, on a visit to the reactive room, no change is detected in the reactive room the AVSA does not query each daemon for its services. It simply accesses the data stored in memory, thus eliminating the confusing delay.

This caching made the system run much more smoothly and thus proved to be extremely valuable. There were other situations in which the system seemed to get stuck, but there was nothing that could be done about them. The reason they were getting 'stuck' is because the media space was slow in responding to requests. The only way to truly fix the problem is to speed up communication within the media space. This is not in the scope of this thesis. Unfortunately, there was no way to work around the problem either because the answer to the request was crucial.

FIGURE 23 The map helps users navigate through the menu hierarchy. The first line of the option list helps the visitor pinpoint their location on the map.

Improving navigation

Some of the users commented that they sometimes felt "lost" when they navigated through the hierarchy. They felt that once they had moved around the media space they no longer had a sense of where they were and how to get to where they wanted to.

To resolve this problem we decided to implement a map as a navigational aid (Figure 23). The map summarizes the menu hierarchy through generalized captions so that a visitor can quickly see where they can go and how to get there. It also indicates the current position through a more specific caption and a different color. To help the visitor further, the current location was displayed as the first line of the option list (Figure 23). In order to ask the AVSA to display the map the visitor must say the word “map”. This word acts as a toggle switch. Consequently, when “map” is uttered again the map will disappear.

Directing speech

In the first prototype, we implemented a simple protocol by which the AVSA knew who the visitor was talking to. Unfortunately, we discovered that the same ambiguity was being experienced by the members of the media space. The problem was that when a visitor was issuing commands to the AVSA they were also disturbing the meeting. It was not so much a problem if it was a one-on-one meeting or even a three person meeting, but with four or more the meeting was more like a presentation and it was disturbing to members of the local media space to have to listen to a person issuing, what seemed like, irrelevant comments.

There are three possible solutions. All involved providing a signal by which the media space knows to mute the audio going to the meeting room, but not the AVSA. The first is to provide a mute button of some sort at the orphan CODEC site. The problem with this solution is that it does not conform to our LCD quality. The second solution was to require the user to use some hand gesture. Unfortunately, this solution is not self-explanatory or easy to use. The visitor would have to learn a special gesture and make sure that they do not use it during the course of the meeting. It may also be distracting to the people in the meeting to see someone performing what seems to be irrelevant gestures. The third solution was to provide the visitor with the command “mute”. When this command is issued the audio to the meeting room, but not the AVSA, was disconnected. When “mute” is said a second time the audio is reconnected. The problem with this solution is that the members of a meeting will still hear “mute” at least once. However, we decided to go with this final option and evaluate it further before considering one of the others.

The help and introduction screens

There were two elements present in most interfaces that were still missing from the AVSA interface. The first was a help screen (Figure 24). The interface is quite self-explanatory. However, we have added two new commands (mute and map). The existence of these commands are not indicated anywhere. We could have displayed them on the visitors screen all the time. This would have been intrusive, especially since we would have to explain their function and usage. We also needed a way for the visitor to know about the "show menu" and "hide menu" commands. As a result, a help screen was added to the interface. To make the help screen visible/invisible the visitor must say "help". Since all the other commands were toggle commands, we also decided to make "show menu" a toggle command so that "hide menu" is no longer needed.

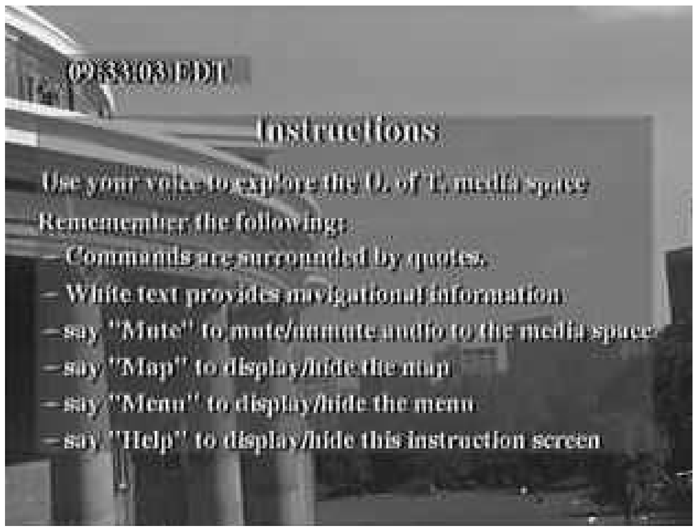


FIGURE 24 The help screen explains how to use the AVSA.

FIGURE 25 The introduction screen.

The next addition to the interface was the introduction screen (Figure 25). Originally when the visitor connected to the media space they saw a list of options overlaid on top of an appropriate sign. This was not very inviting. Instead, we redirected the video seen by the visitor to be that of a live view of the University of Toronto. We then overlaid a greeting on top of this image that explained to the visitor where they had connected to, how to get help and the options that were available.

In summary, this section explained the many new interface features we added and changed, based on user feedback, to make the interface more effective. The next section describes the features added to the system to make it more useful.

5.4.2 Adding system features

During this stage we improved the communication component of the system. As a result we were also able to add some new features to the system that make it more useful. The following sections discuss these features.

Rooms with services

The reactive room subsystem was being developed in parallel with the AVSA. One of the developments was the deployment of another room with services. It was desirable for the AVSA to recognize this fact and act accordingly. As a result, we gave the AVSA the ability to recognize any room that has services by simply registering the new room and all the possible node names in a database. When a visitor chooses to call this node the AVSA checks the database and, if the room is registered, it queries the room's local daemons for services.

Multiple visitor access

Within a room it was possible to have multiple visitors. The first prototype system did not address this functionality. It only gave CODEC visitors access to one of the possible visitors to a room. Problems arose when a member of the media space was visiting the room through the TP application and they were visiting as the visitor that would normally be occupied by a CODEC visitor.

In this situation the CODEC visitor would be able to complete a connection to the room but they would not be able to access the room services. This stage addressed this problem so that as long as there was a node available in the room, the visitor would be able to access any of its services.

Video on demand

So far, the only advantage we have taken of the television technology in our effort to converge technologies is that we are exploiting the richness of the medium through which it is being transmitted. However, as mentioned in chapter 1, the concept of the television as a method of distributing information at any time is also very valuable.

For this reason we decided to implement a video-on-demand (VOD) service. With this service visitors can access pre-recorded material at any time. VOD also enables members of the media space to allow visitors to access whatever material they wish them to access.

Not only can visitors access the information, but they can pick what they want to see from a list of demos (Figure 26).

FIGURE 26 The user has access to various demos provided by members of the media space.

The AVSA also gives them such control as play, stop, pause, fast forward, fast forward speed, etc. (Figure 27).

FIGURE 27 The VOD extension allows one to control viewing of the demo.

5.5 Summary

During this stage we moved the AVSA from the PC/Windows platform to the SGI/UNIX platform to make the system more stable and expandable.

We used comments provided by users at the end of the second stage to improve the overall look, feel and effectiveness of the interface by improving and speeding up navigation, improving feedback, making the interface more visible, introducing the system, etc.

New functionality in the form of video-on-demand, access to other rooms with services, and access to multiple visitors within a room were added to improve the quality of the visit to the media space.

CHAPTER 6 Future Considerations and Final Words

This chapter presents many issues that need to be addressed by future efforts to develop the AVSA. We also summarize the work and present the major contributions of the work.

6.1 Future Considerations

We would like to see the AVSA deployed into the field at some point in the future. The development and testing of the AVSA system has raised a number questions which could not be fully addressed within the scope of this research. However, they are issues that will have to be dealt with in order to make the system deployable. We therefore present these issues and possible solutions in the following sections.

6.1.1 Privacy

Reviewers of the system pointed out that they would not like people to have the ability to "roam around the media space, visiting offices indiscriminately". They felt this was a violation of their privacy[5][11][19]. Let us look at the current situation through an example.

Imagine the case where you are being visited and someone is trying to contact you through the AVSA. Currently, if your node is busy with a visit from another electronic visitor, the new visitor is not allowed to connect to you. This is fine if the node is busy. In this case, a "node is busy" reply is sent from the media space to the AVSA which then provides a message in the banner to the visitor saying "The person/room you have called is busy. Please try later."

However, what if your node is not busy? The TP application allows one to specify whether or not electronic visitors can connect to one's room. It uses the a door[6][20] metaphor whereby an open door means allow electronic visitors to enter the room and a closed door signifies that the room is not open to electronic visitors. There are four possible scenarios that could occur with an AVSA enabled media space.

The first is that your door is closed on purpose. In this situation the media space sends a busy signal to the AVSA which relays this message on to the visitor. Hence, the visitor can conclude that you are busy and he/she will try again later. This situation is acceptable as it enables privacy and the visitor is satisfied in knowing that you are busy.

However, what if your door is closed by mistake? Once again the media space sends a busy message to the AVSA. The visitor erroneously concludes that you are busy, when in fact, for example, you have simply forgotten to open the door after closing it earlier when you did not want to be disturbed. This situation is inappropriate. Privacy is valuable, but so is the ability to communicate on demand. The effectiveness of the system is hampered by situations like this because both you and your visitor are missing the opportunity to communicate valuable information.

The third scenario is if your door is open on purpose. If you are willing to accept a visit from anyone or the person you are waiting for a visit from connects to you then there is no issue of privacy. However, if someone other than the person you are waiting for calls first you must decide if this visitor's visit is more important. It is entirely possible that it is not, but during this decision process the visitor you are waiting for may call and may be turned away by the system.

The fourth scenario is that your door is open by mistake. This situation is very dangerous as it could lead to leaking of important documents, embarrassing situations, meeting disruptions, etc. Privacy in this scenario seems to be the major concern of reviewers of the current AVSA.

Taking into the consideration all of these problems, it seems that the best solution would be to borrow the proven protocol used in telephony. In short when a call is made to you through the AVSA, the AVSA instructs the media space to send a special signal to your node that notifies you that someone wants to meet you. If you are in your office you can accept the call. If not the ringing will continue for a pre-specified time. Once the ringing has timed out the media space will send the busy signal to the AVSA.

A further advancement on this system would be to implement the equivalent to a "call-display" like feature. For example, a view of the visitor could be routed to you so you know who is trying to visit you. Some visitors may take exception to this as people are more comfortable with being able to see what can see them. Therefore, an alternative solution is to prompt the visitor to provide some audio, like saying their name. This audio would be sent to you in the form of, for example, "ring - incoming visit from 'Bill Buxton' - will you accept?". This way you would have the information necessary to decide whether or not to receive the visit. At this point the system could even be designed to wait for a verbal "yes" response from you to accept the call.

This solves all of the problems mentioned earlier, but it introduces one that is not in the current system. That being in the current system you do not have to make any special effort to receive calls. To preserve privacy there is no way to totally eliminate this inconvenience. However, we can propose a method of specifying situations that will allow members of the media space to reduce this inconvenience.

6.1.2 Different levels of access

It is clear from the discussion on privacy that we need some way of specifying different levels of access. It makes sense to base this specification on social proximity, because closer people have different rights than people who are further away. It also makes sense to base the accessibility on the situation. For example, a meeting may have a different accessibility rule than visiting a personal node. Unfortunately, the only person who can truly define the level of closeness and can evaluate the situation, in our context, is the person of the media space who is being visited. This is essentially the situation in the caller-id system which was previously discussed. The person being visited evaluates their current situation and makes a decision as to the importance of a call from a visitor based on social proximity.

One way to address this situation is to use a human receptionist to filter calls as they come in. The receptionist will take the appropriate action depending on the situation and social proximity. By and large this non-technology solution will work, but it is interesting to note that the receptionist will be basing the decision on a model they have learned through experience.

Our goal with respect to this issue is to narrow the possibilities down as much as possible and construct a simple model, similar to one that a human receptionist may use, upon which the AVSA can determine whether or not the visitor should go through the caller-id system. The criteria which we will use in this accessibility model are obviously social proximity and situation.

There are two main social groupings of people who can be detected by the AVSA. The CODEC visitors and the TP application visitors. In the current system there is no way to distinguish one CODEC visitor from another. On the other hand internal media space users can be distinguished from each other much more easily based on the their node id (we are assuming, perhaps dangerously, that only the person logged on to the node will be using that node). Hence, internal users can be further divided into social groupings. It should be noted, however, that as speech technology develops it will be possible to distinguish one CODEC caller from another using speaker verification and/or speaker identification technology. This technology will also help enforce our assumption that, for internal media space users, the person logged on to a node will be the person using the node.

Imagine a graduate student specifying level of access based on his/her view of social groupings. A graduate student may require five groupings. One for CODEC visitors, one for students, one for people he/she is supervising, one for peers and one for his/her supervisor. The graduate student must place these groupings into levels that determine the accessibility of the graduate student's node. These levels could be:

1. no access (door locked)
2. access through caller ID process only (door closed)
3. glance into room any time, but access with caller ID (door partially open)
4. glance into room any time and access any time (door fully open)

CODEC visitors may be assigned to level 2, which means they must go through the caller ID system before they can complete a connection. On the other hand the graduate student could assign her/his supervisor the highest level, 4, meaning they value the opportunity of the meeting so much that the supervisor is always welcome. Level 3 could be assigned to people being supervised by the graduate student and so on. This type of grouping system could be made available to the owner of the node as properties which can be set so that in certain situations the node owner could define who should be allowed what level of access.

The first example dealt with assigning access levels based of the social proximity relative to the owner of a personal node. However, what if the node is a meeting node? For this type of node it does not make sense to assign social groupings. The room is meant to be a shared space where anyone can meet and exchange ideas. Level of access will therefore be determined by the situation at hand. Once again there are some distinct situations that can be identified. These situations could be used as a description of the level of access to the room. For example the description could be:

- a highly confidential meeting - meaning only physical attendees are allowed.
- an internal meeting - meaning must use caller ID.
- an open meeting- meaning allowed to glance and enter at will but must be introduced upon entry.
- a demonstration or nothing going on - meaning enter at will, no introduction required.

Once again, these situational specifications can be provided as properties of the room to be set when a meeting is to take place.

In summary, the settings of the two criteria could be made available to the nodes of a media space so that intelligent decisions can be made about the level of access people should have. The AVSA could even be developed further so that it allows the visitor to make intelligent decisions on how to enter the media space. For example, if the visitor is looking for someone and they decide to visit nodes asking for the person. The visitor would have reservations about entering meeting rooms and possibly disturbing a meeting. By being aware of what is going on in meeting room the AVSA can inform the visitor of the situation by displaying a message like “This room is being used for an internal meeting. Do you wish to wants to enter? Yes or No”. The visitor can then make an intelligent decision as to the importance of their need compared to the situation in the meeting room.

6.1.3 Video Mail

In dealing with the privacy issue we have uncovered a functional deficiency of the system which should be addressed. In all situations where the visitor attempts to connect to a member of the media space and is turned away they must try to call again later. There is no way to provide information to the member of the media space that will allow him/her to pass on a message like who called, how they can be reached and when they can be reached. Nor is there a way for an orphan CODEC user to simply leave a message for someone at a media space.

To address these problems a video mail feature should be integrated into the system. It would be invoked in two ways.

- If the visitor tries to call a node that is busy. Thus, the video mail acts as an answering machine.
- From the main menu if the visitor just wants to leave a message for a member of the media space. In this case the video mail acts as an e-mail system.

6.1.4 Directing speech

The question of how to direct speech to the AVSA and how to direct speech to the media space needs to be addressed further. Thus far the AVSA knows if a visitor is talking to it based on whether or not the menu has been activated through the “menu” command. If the visitor has connected to a member of the media space they have the option of muting the audio going to the media space first by uttering “mute” so that they do not disrupt meetings.

Unfortunately, muting the audio in this way is still too intrusive to large presentation-like meetings in the media space. A non-auditory signal is required. The two other possible solutions we presented earlier were to allow the visitor to use a gesture to mute the audio or to give the visitor access to a mute button of some sort. The former solution forces the visitor to learn a gesture that may or may not make sense or even be appropriate (depending on cultural differences). It also requires that the visitor be aware of his/her position relative to the camera. Finally, it may disturb other meeting attendees.

The solution seems to be to provide a mute button without adding any extra equipment at the orphan CODEC. The best way to do this is to write a simple audio monitor that monitors the level of audio being received from the CODEC. When the visitor wants to talk to the AVSA without first disturbing the meeting, the visitor can turn the audio output of their microphone off first. The audio monitor will sense this and the AVSA will mute the audio going to the meeting room. The AVSA can then display a message to the visitor through a banner saying that the audio has been muted. Any audio from now on will be directed toward the AVSA. The visitor can then turn the audio output of their microphone back on and talk to the AVSA. When the visitor wishes to have their audio routed to the media space again they can issue the "mute" utterance to toggle it back on. The solution is quite natural, it is non-intrusive, and it does not require extra equipment. The only drawback is that it does require the visitor to have access to the audio level (i.e. volume control) coming out of their microphone.

6.1.5 Controlling Volume

In almost every meeting which is attended by an electronic visitor the electronic visitor, the electronic visitor has had to request the member of the local media space to adjust the volume being sent to the visitor. There is no reason why electronic visitors should not have the ability to adjust the volume themselves.The implementation would require some centrally located hardware and software control that communicated with the AVSA using the established protocol. Volume control would then be added as a service of the room through a room control system.

6.1.6 Speech Recognition

Up until now we have designed the AVSA for discrete-utterance, small-vocabulary, speaker-independent speech recognition. Because of the dynamic nature of the media space we feel that the discrete-utterance menu-guided approach we have taken is a very effective one. However, there are some situations in which performance and naturalness could be improved by using a continuous-utterance recognizer with word-spotting.

For example, when the user wants to see the menu it would be much better if the visitor could say any one of "Attendant, please show me the menu" or "Attendant, show me my options" or some other variation of the same command. By not restricting the user to a specific command it would allow users to *talk* to the system in a more natural manner. An added benefit is that erroneous recognition, due to similar sounding words, would be greatly reduced.

6.1.7 Increasing Access

In the past, access to a media space has been limited to members of other media spaces. Now, with the AVSA, people with orphan CODECs can access a media space as well as its services. While we have significantly increased the ability to communicate it should be noted that many people do not have the expensive equipment required to take advantage of this technology. However, by developing the system as a set of independent layers we have made it possible to extend accessibility to people who want to access the media space through some other means. Two possibilities are:

- telephones
- public access kiosks

Telephones

We know that video-enhanced communication improves the quality of the communication[25] and thus telephone based communication will not be as effective in the context of videoconferencing. However, we cannot ignore the fact that telephones are one of most ubiquitous communication technologies available to us today- more than traditional videoconferencing sites and much more media spaces. For this reason, it is extremely important that once deployed the AVSA should be able to accommodate telephones.

The layered architecture of the AVSA does in fact make it possible for accessibility to be extended to telephones. Instead of using the video channel to present the options to the visitor, the options would be presented using the audio channel. This would be facilitated by adding a text-to-speech synthesizer to the AVSA. The AVSA would simply pipe parsed text from the media space into the text-to-speech synthesizer. The resulting audio would be sent as options to the visitor. Input would be conducted through the audio channel as before or using the standardized touch-tone keypads along with DTMF signals.

Public access kiosks

There are currently no public access kiosks that connect to a media space. It would not have made sense to provide them in the past because without the ability to navigate through the media space and control the media space it would not have been useful or cost-effective. The AVSA solves this deficiency and it now makes sense for kiosks to be made available to further increase the ability to communicate.

As mentioned before, the AVSA was developed in layers. So once again the interface layer can easily be changed to accommodate any other type of interface. For example, the kiosk may be in a noisy environment and work better with a touch screen interface.

6.2 Summary and Conclusions

We identified the lack of external access to a media space as a serious deficiency in media space communication. We propose a solution to the problem in the form of a system called the AVSA. The AVSA enables access and control of a media space from a traditional videoconference room.

In a larger context, we explored the concept of using various technology to facilitate communication efficiently and effectively. We note, however, that all these technologies have properties that detract from their purpose. We also note that to combat these deficiencies the different technologies are being developed so that they *mimic* properties of the others. However, because of the approach, they are limited by the limits of the technology within which they operate.

We proposed to explore "convergence" of technologies from a different perspective. One that captures the advantageous properties of each to reduce the disadvantageous properties of the others *without* centering the development around any one technology. For our exploration, we chose to augment a media space, the Ontario Telepresence Project, with a system called the AVSA.

One key issue is that we had no control over the videoconferencing room. We had to work with the equipment that was already there- a camera, microphone, monitor and speaker. The resulting interface enabled control to be exercised through speech input in response to speech prompts.

The second key issue was that if the work was to have any influence, as an example, on efforts to converge technologies we had to develop a useful system in a timely manner. As a result, we developed the AVSA through a thoughtful, three-stage, iterative process involving input, on an informal level, from users, developers and reviewers of varying backgrounds.

Issues of privacy, access, added functionality, etc., which will require attention before the system can be deployed into the field for public use, have also been discussed along with suggested solutions.

From the comments of people who have evaluated the system we concluded that we had significantly increased media space access and contributed to the discourse on networked interactive information appliances; thus, achieving our objectives.

6.3 Contributions

Functionally, the AVSA is significant in two ways. First, it increases media space accessibility by:

- enabling people from traditional videoconferencing rooms to access the media space.
- providing an alternative, and potentially easier, means by which members of other media spaces can access a media space.
- providing an easily modifiable system, through its layered architecture, that enables other devices, like the telephone, to become the technology through which interaction can take place.

Second it provides an interface through which people can control media space resources that were previously not controllable through any interface, making the visit more effective.

The AVSA is also significant in the following ways:

- it provides an example of an alternative option to the conventional perspectives on how to converge technologies.
- because of the constraints rooted in our problem, we were forced to build an interface with unusual modalities. This work serves as an example of how to work with similar constraints to build an effective interface.

References

1. Anderson, Rick, “Interactive Television - The Excitement Awaits,” *High Technology Careers*, 1994.
2. Arons, B., “SpeechSkimmer: Interactively Skimming Recorded Speech,” Proceedings of UIST’93, November 3-5, 1993, pp. 187-196.
3. Bly, S., Harrison, S. & Irwin, S., “Media Spaces: Bringing People Together in a Video, Audio and Computing Environment,” *Communications of the ACM*, 36(1), January 1993.
4. Buxton, W. & Moran, T., “EuroPARC’s Integrated Interactive Intermedia Facility (IIIF): early experience,” In S. Gibbs & A.A. Verrijn-Stuart (Eds.). Multi-user interfaces and applications, *Proceedings of the IFIP WG 8.4 Conference on Multi-user Interfaces and Applications*, Heraklion, Crete. Amsterdam: Elsevier Science Publishers B.V. (North-Holland), 11-34.
5. Buxton, W., “Social Planning and Communities Along the ‘Information Highway’”, Interactivity, 1(1), 85-88. Also appears in Japanese in: Buxton, W. (1994). Social planning and communities along the “Information Highway,” InterCommunication, 10. Tokyo: NTT, 168-171.
6. Buxton, W., “Ubiquitous Media and the Active Office.” Published in Japanese (only) as, Buxton, W. (1995). Ubiquitous Video, Nikkei Electronics, 3.27 (no. 632), 187-195.
7. Cooperstock, J.R., Tanikoshi, K., Beirne, G., Narine, T., Buxton, W., “Evolution of a Reactive Environment,” *Proceedings of CHI’95*, pp. 170-177.

8. Danis, C. & Karat, J., “Technology-Driven Design of Speech Recognition Systems,” *Proceedings of DIS’95*, Ann Arbor, MI, USA, pp. 17-24.

9. Gaver, W., “The Affordances of Media Spaces for Collaboration,” *Proceedings of CSCW’92*. November 1992, pp. 17-24.

10. Gaver, W., Smets, G. & Overbeeke, K., “A Virtual Window on Media Space,” *Proceedings of CHI’95*, ACM Conference on Human Factors in Software, pp. 257-264.

11. Gaver, W., Moran, T., MacLean, A., Lovstrand, L, Dourish, P., Carter, K., and Buxton, W., “Realizing a Video Environment: EuroPARC’s RAVE System,” *CHI’92 Conference Proceedings*, pp. 27-35.

12. Grudin, J., “Why CSCW applications fail: Problems in the Design and Evaluation of Organizational Interfaces,” *Proceedings of CSCW’88*, ACM SIGCHI & SIGOIS, Portland, OR, Spetember 26-29, 1988, pp.85-93.

13. Gujar, A., Daya, S., Cooperstock, J., Tanikoshi, K., and Buxton, W., “Talking Your Way Around a Conference: A speech Interface for Remote Equipment Control.” *CD-ROM Proceedings of CASCON’95*, Toronto, Ontario, Canada. Also available from http://www.dgp.toronto.edu/people/anujg/papers/casconPaper.html

14. Gujar, A., Tanikoshi, K., Cooperstock, J., Daya, S., Buxton, W., “Combining Technology Through The Audio/Video Server Attendant,” Unpublished manuscipt, available from http://www.dgp.toronto.edu/people/anujg/links/chi96long.html

15. Gujar, A., Cooperstock, J., Daya, S., Buxton, W., “Eliminating the Hole in Media Space Communication,” Unpubished manuscipt, available from http://www.dgp.toronto.edu/people/anujg/links/chi96short.html

16. Harrison, B., Vicente, K., “An Experimental Evaluation of Transparent Menu Usage,” *Proceedings of CHI’96*, Vancouver, B.C., Canada, pp. 391-398.

17. Harrison, B., Ishii, H., Vicente, K., Buxton, B., “Transparent Layered User Interfaces: An Evaluation of a Display Design Space to Enhance Focussed and Divided Attention,” *Proceedings of CHI’95*, Denver, Colorado, USA, pp. 317-324.

18. Mack, R., Nielson, J., “Usability Inspection Methods: Executive Summary,” *Human-Computer Interaction: Towards the year 2000*. Edited by Baecker, R. et al., Morgan Kaufmann Publishers Inc. 1995, pp. 170-181.

19. Mantei, M., Baecker, R., Sellen, A., Buxton, W., Milligan, T., and Wellman, B., “Experiences in the use of a media space,” *Proceedings of CHI’91*. ACM Conference on Human Factors in Software, pp 203-

208. Reprinted in D. Marca & G. Bock (Eds.) 1992. *Groupware: software for computer-supported collaborative work*. Los Alamitos, CA.: IEEE Computer Society Press, pages 372-377.

20. Moore, G., “Videoconferencing 1990s Style: Sharing Faces, Places and Spaces,” Ontario Telepresence Project Technical Report, 1994.
21. Resnick, P., “HyperVoice: A Phone-Based CSCW Platform,” *Proceedings of CSCW 1992*, November 1992, pages 218-225.
22. Riesenbach, R., “The Ontario Telepresence Project,” *CHI’94 Conference Companion*, pages 173-174.
23. Schmandt, C., “An Audio and Telephone Server for Multi-Media Workstations,” *Proceedings of the 2nd IEEE Conference on Computer Workstations* 1988, pages 150-159.
24. Schmandt, C., “Phoneshell: the Telephone as Computer Terminal,” *Proceedings of ACM Multimedia* 1993, pp. 373-382.
25. Tang, John C. and Ellen A. Isaacs. “Why Do Users Like Video?” Studies of Multimedia-Supported Collaboration. MountainView, CA: Sun Microsystems Laboratories, Inc. [Technical report, Dec. 1992]
26. Wattenbarger, B., Garberg, R., Halpern, Lively, B., “Serving Customers with Automatic Speech Recognition - Human-Factors Issues,” AT&T Technical Journal, May/June 1993, pp. 28-41.
27. Weiser, M., “The Computer for the Twenty-First Century,” *Scientific American*. September, 1991. pp. 94-104.
28. Zellweger, P., Terry, D., Swinehart, D., “An Overview of the Etherphone System and its Applications,” *Proceedings of the 2nd IEEE Conference on Computer Workstations*, 1988, pp. 160-168.