Enhancing Network Services

through Multimedia Data Analysers

Ferdinando Samaria, Harold Syfrig, Alan Jones and Andy Hopper
ORL, Olivetti & Oracle Research Laboratory
24 Trumptington St
Cambridge CB2 1QA
United Kingdom
+44-1223-343 000
ahj@cam-orl.co.uk

ABSTRACT
This paper summarises our experience of using data analysers to enhance network multimedia services. By analysers we mean simple processing modules that extract information contained in various multimedia streams. They are categorised based on their location with respect to the network, the location being determined by balancing bandwidth requirements and computational complexity. Various applications are described where analysers are used to enhance aspects of the service provided. Details of the multimedia environment are given, followed by an overview of the analyser architecture and examples of analyser-enhanced applications. The paper is concluded by indicating directions of future development.

KEYWORDS
Data analysers, distributed multimedia, ATM network services.

INTRODUCTION
Advances in network technology permit sophisticated configurations in the local area, which have resulted in a wealth of multimedia data being exchanged on the network. It has therefore become necessary to establish a transfer medium that can accommodate both traditional computer data traffic (e.g. bursty file transfers) and communications data (e.g. continuous audio and video). Asynchronous transfer mode (ATM) [4] goes some way towards presenting a unified network that supports both kinds of data. ATM is a connection oriented protocol based on small fixed-size cells, hence enabling fine grain cell preemption and low latency.

As the cost of peripherals capable of generating multimedia data (e.g. cameras and microphones) continues to drop, it is reasonable to expect that such peripherals will be commonly used at the desktop. The traditional way to interface these devices to the network is via specialised boards on the workstation's I/O bus. An alternative approach moves the devices from the bus and attaches them directly to the network [1, 7, 16]. In this architecture, the networked devices can be easily shared, the I/O intensive work is moved away from the workstation and it is possible to set up, for example, audio only locations without the need for an expensive workstation.

The platform used throughout this paper is the Medusa system [6, 16], which is based on devices that attach directly to a 100-Mbit ATM network. Each device is an independent computer based on a 32 MHz RISC microprocessor. The devices run ATMos, a lightweight operating system developed at ORL. The peripherals include video [3], audio, compact liquid-crystal display and storage devices [2]. Figure 1 illustrates a typical hardware configuration for the Medusa system. The video devices can generate video at six simultaneous resolutions, according to the needs of an application. For example, a video-conferencing application uses high resolution images, while a simple motion analyser can operate on a small resolution image to save computing cycles. The audio devices can sample data at up to 48 KHz, yielding very high quality audio. Also present on the network are storage units that can hold up to 16 GBytes, and general purpose computers, where demanding computation can be carried out. Two categories of such computers are presently available: processor banks based on collections of ARM processors and Alpha workstations. The ATM equipment used was developed at ORL and is now being commercialized by ATML (''Advanced Telecommunication Modules Ltd'').

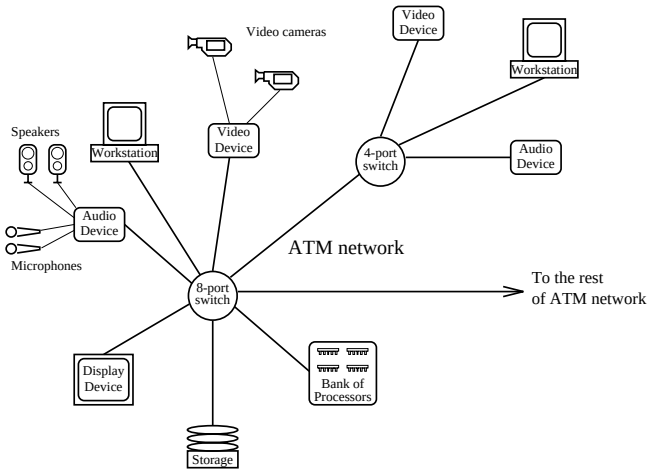

Figure 1: Direct access devices

This type of architecture, where the peripherals are directly plugged into the network instead of being attached to a workstation's internal bus, allows us to have a great number of them present on the network. The system scales very well and all these devices are shared. This led to a set of distributed network services using numerous simultaneous streams being available to Medusa users. Some had been experimented with in a previous system developed at ORL called Pandora [8], which included multimedia services like videophone and videomail [12]. However, Pandora was concerned primarily with data transport, with little emphasis on being able to access the data for the purpose of analysis. Medusa, on the other hand, was designed to allow access to the data as it flows through the system. In Medusa, it is simple to send streams to software units ("analysers") developed to carry out specific data processing tasks. Through the use of analysers, some of the basic multimedia applications have been enhanced. Analysers can be useful to manage multiple data streams (for instance by selecting the active microphone in a room containing a multitude of multimedia peripherals) and introduce a range of new applications where certain functions or user actions can be automated. This paper describes a set of applications which make use of analysers. In the following sections, the general overview of the Medusa environment, the analyser architecture and some applications will be presented. The paper is concluded by summarising our experience and outlining the future directions of work in this area.

OVERVIEW OF THE MEDUSA SYSTEM

Medusa provides a networked multimedia environment, based on a collection of direct ATM peripherals and distributed software objects called *modules*. A server process, running on each ATM device, contains special modules called *factories*, which can create other modules. The servers are specific to the device and can run under UNIX, Windows NT or ATMos. Modules communicate with one another through reliable unidirectional connections. Data flows through connections in atomic units called segments, which are grouped in data transfer units called messages. A module's internal state is exposed through a set of attributes, which can be read, written or notified to other modules. Connections are network transparent and modules can in principle reside on any of the hardware devices. One of the aims of Medusa is to make the manipulation of multimedia data simple and analyser modules can be easily added to data pipelines.

THE ANALYSER ARCHITECTURE

Analysers can be characterised through various features. For example, they can be categorised based on their function and the level of abstraction of the function they perform. In this case, video analysers would range from simple motion detection to gesture recognition and scene understanding analysers. In this paper analysers are categorised based on their location with respect to the network. It is the analyser's complexity and the type of function it carries out which determine where the analyser is located. Analysers are divided into three categories: source analysers, sink analysers and network analysers. Source analysers act on the data before it has traveled onto the network, while sink analysers act on data that has traversed the network for purposes other than analysis. Network analysers are processing services that can run on general purpose computers on the network and are available to all applications. The three categorises of analysers are illustrated in figure 2.

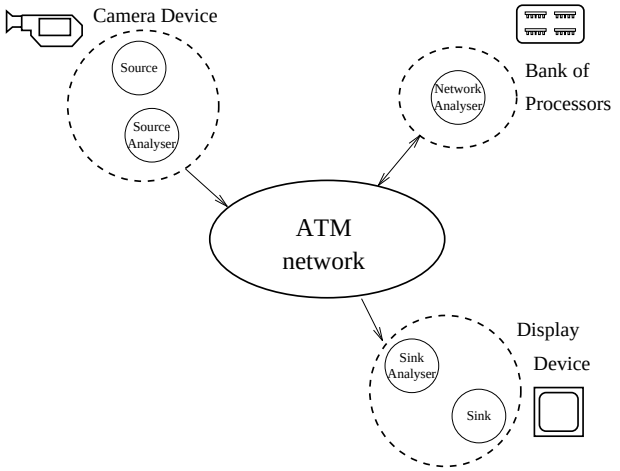

Figure 2: Source, sink and network analysers

Applications which require complex processing are normally implemented through a set of analysers. The analysers are arranged hierarchically, where simple tasks are carried out by first-stage analysers. The first-stage analyser is computationally light-weight and normally processes low resolution data streams continuously. When

an event of interest is detected, it triggers the next-stage analyser, which either works on higher resolution data or performs a more complex task, and so on. A simple example is the gesture recognition application which will be described later in this paper, where the hand-tracking analyser becomes active when a first-stage, simple motion analyser signals a motion event in the stream.

Source analysers

Source analysers reside on the same device as the data source module and run within the same process. Generally, source analysers are used to carry out simple processing, where the cost of the computation is low compared to the extra network resources required to send the data to a network analyser. They are also used when information is to be shared across a number of streams (e.g. time, loudness and motion stamps). It makes sense to locate certain types of simple analysers at the source, such as compression and annotation analysers.

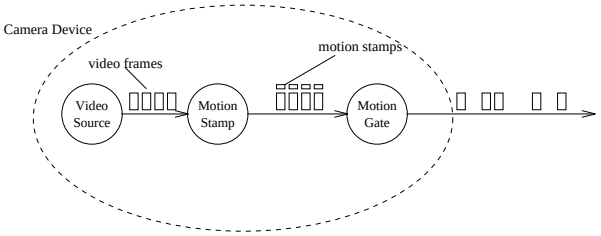

Figure 3: The motion stamp & motion gate modules

A common type of source analyser entails stamping on the outgoing stream annotations which describe the data content, for example the motion level in a video frame or the loudness level in an audio block. The stamping is performed at source to facilitate sharing of the information if multiple sinks are present. Figure 3 shows an example of motion stamping on a continuous video stream, where motion is the percentage of pixels that have changed between the current and the previous frame. The motion stamping module can be combined with a motion gate module at source, which lets through only those frames whose motion stamp is above a certain threshold. This can be used as a simple video compression scheme, where only those video frames which are substantially different from the previous are transmitted. The compression is observed when similar successive frames are discarded.

Sink analysers

Sink analysers share the same device and process as the data sink module. Sink analysers can be used to relieve the source from carrying out functions that may be specific to the sink device. Sink analysers perform simple processing that would not justify the extra network traffic generated by sending the data to a network analyser. Such analysers include converters, which process the data to a format suitable for the sink device, and decompression analysers. In a pipeline of compressed data, analysers which process uncompressed data are also generally located at the sink, as the decompression module itself normally resides at the sink. Other sink analysers include those which read annotations off an incoming stream and take simple decisions based on the information contained in the annotation itself. Examples of such analysers will be given in the following sections.

Network analysers

Network analysers are modules that provide data analysis services and are normally located on general purpose computers. Such devices have superior computing resources and are designed to process multiple data streams. In this case, the benefits of moving the computation away from the source and sink devices to a more capable device outweigh the extra network resources required to send the data to the network analyser. The examples of network analysers used in Medusa include data correlation analysers (optical flow estimators based on block matching [13]) and active stream switches. More complex data services are also available through network analyser. For example, a pattern recognition analyser based on Hidden Markov Models [10] has been used to provide experimental speech [14] and face recognition [11] services.

APPLICATIONS

This section describes various applications which make use of analysers. The applications comprise enhanced versions of standard services such as video-phone and video-mail, and new applications entirely based on analysers.

Active stream selector

Unlike other standard multimedia systems, the video-phone and video-mail applications in Medusa are based on multiple streams, because offices at ORL are equipped with multiple microphones and cameras, typically four of each. The selection of the stream of interest can be either carried out manually or triggered by an active stream selector. The selector can handle audio and video equally. It is based on a token-passing model, in which contention is resolved by comparing streams through some metric and passing a single token to the stream of interest. Interest is assessed using score functions appropriate to the data type.

For example, the score function for video is based on motion, while the one for audio is based on signal loudness. The active stream keeps the token until it becomes inactive, at which point an election is made to determine the new active stream, if any. The stream selection is achieved through a combination of source and network analysers.

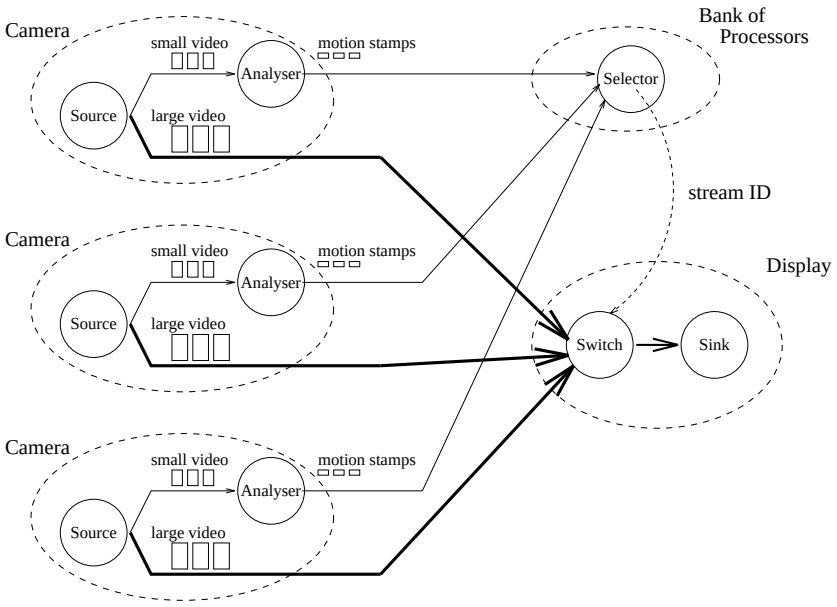

Figure 4: The stream selector

The source analysers create a motion stamp stream which is sent to a network selector module. The selector compares the incoming streams, selects the active one and sends the identity of the winner to the switch located at the sink. The block diagram in figure 4 shows the module arrangement for the video selector. In the Medusa implementation, once one of the four streams is selected, only that stream is active and no data flows down the other three connections. When using video-phone and video-mail in Medusa, the user is potentially free to move around the office during a conversation or a recording, provided microphones, cameras and display devices have been distributed sensibly to cover the areas of interest. This results in a system in which the user has relatively few restrictions in his or her movements.

Simple video compression

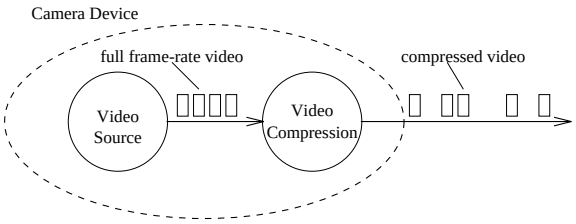

Figure 5: Video compression at source

A scheme, based on the same principle as that illustrated in figure 3, can be implemented more effectively as shown in figure 5. In this case the stamping and gate modules are combined into one single compression module, also resident in the source device. This solution can be more efficient because the compression module can stop differencing image pixels as soon as the pass threshold is exceeded.

Also, with the new scheme it is possible to compute the difference between the current frame and the last frame sent, which is more suited for detecting slowly changing scenes (in the other case, a slowly changing scene would result in a sequence of low motion stamps, as the stamper compares each frame to its previous and no frames would be sent). This is illustrated in figure 6.

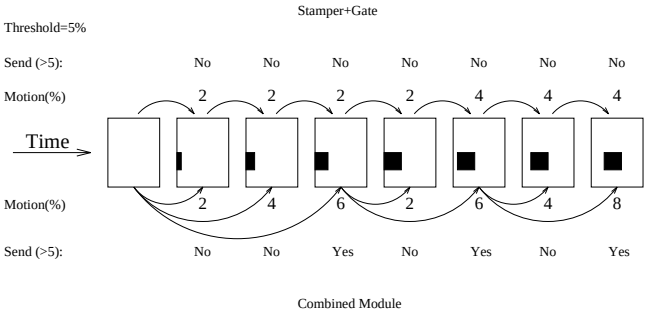

Figure 6: Stamper + gate vs combined analyser

This video compression scheme is used in our multistream video-phone application. When a connection is established between two locations, multiple views are transmitted and received, as two or more cameras are present at each location. All available views are presented simultaneously to the user, with the one of interest magnified as in figure 7, where an example with four cameras is shown. Normally the scene changes only in the field of view of one or two cameras, with the rest effectively using up little or no network resources. This is the case particularly for

cameras pointing at desks, whiteboards or bookshelves.

Figure 7: Presenting multiple streams to the user

Gesture recognition

Gesture analysers have been implemented to experiment with novel user interfaces driven by hand movements. Two sets of analysers have been implemented. The first is a complex network analyser based on active contour models [5], in which a virtual glove (as seen in figure 8) tracks hand movements in real time, recognising a limited vocabulary of gestures. The network analyser is activated by a source analyser which signals every time sufficient motion is present in the video stream. This saves the network analyser from continuously computing estimates of the hand position. The second is a simple source analyser that turns a video camera on and off by detecting a hand wave close to the camera lens. The hand wave is characterised by a large motion signal, which can be easily separated from normal office activity.

Archiving and retrieval

A simple application based on a video camera pointing at a whiteboard was used to archive high-resolution snapshots of the whiteboard when differences were detected. The snapshots were stamped with information such as amount of change, date and time. The application consisted of a source motion analyser and a fine grain motion network analyser, used to compute differences between a reference frame and the current one. When motion was observed by the source analyser, the reference frame was set to the last frame seen before motion had commenced. By changing the reference frame in this way, it is possible to cope with slow light changes which occur during the day. The network analyser would become active as soon as the source analyser would see no further motion and would compute fine grain differences between the reference image and the current one. The images would be tiled into a number of smaller resolution blocks and differences would be computed on a per tile basis. If sufficient changes were observed, the current frame would be stamped with the appropriate information and sent to a file store unit.

Figure 8: The hand-tracker

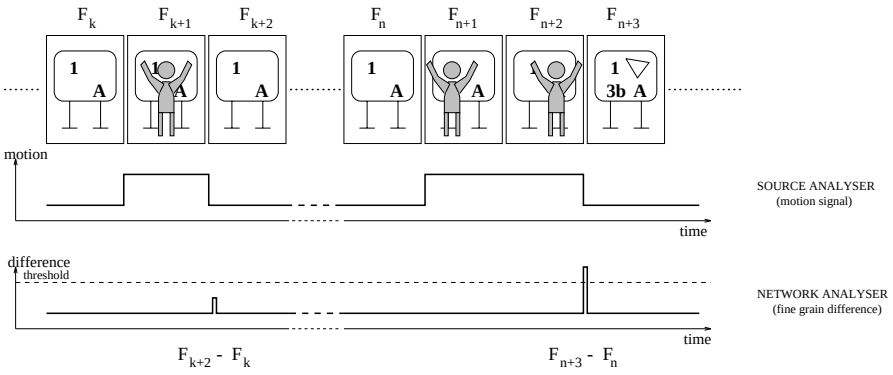

Figure 9: Timing diagram for the whiteboard application

Using this application, it is possible to build an archive of whiteboard snapshots, which should only comprise those where interesting changes have occurred. Figure 9 illustrates a possible timing diagram for the application. The frames show a sequence where a person appears in front of the whiteboard twice. The first time, no changes are made to the whiteboard. The second time, new characters are written to the whiteboard. The source analyser activates twice, when the person appears in the picture. The network analyser is activated as soon as motion in the source analyser goes back to low. At this point, the reference frame (the last frame before motion commenced) and the current frame are compared and if fine grain changes above a set threshold are detected, the current frame is archived. The first time, not enough fine grain changes are detected (i.e. motion is not above the threshold level) and the frame is therefore not stored. Typically, this happens when a person walks past the whiteboard. The second time enough changes are observed and the frame is stored.

Security and monitoring

One area where software analysers are particularly suited to assist human operators is that of security and data monitoring. A security application has been implemented, which opens a small-size video stream from every known camera on the network and forwards the data to a collection of network analysers. If movement and sound are detected in the image, the video stream is magnified to reveal finer details. The application is illustrated in figure 10. A similar application was run over periods of holiday coupled with Active Badges [15], a system which locates people around the building. If movement was detected and no Active Badge was detected at the scene, the video data was sent to a remote file store for further inspection.

Other applications based on network analysers enable monitoring of audio levels and computing optical flow through block matching in a live video signal. The sound analysers can be used, for example, to avoid interrupting people in a meeting or on the phone. The optical flow analyser can be used to augment a location system like the one based on Active Badges, by locating a single person in the field of view of the video camera.

CONCLUSIONS

The material presented in this paper is the result of extensive experiments within a modular, high-speed, networked multimedia environment. The flexibility offered by the Medusa system provides an ideal platform for the development of new, distributed, analyser-driven applications, and much experience has been gained throughout the previous years in handling multimedia data in such an environment [8]. Starting from a hardware infrastructure comprising over one hundred distributed end-points, we have studied and implemented many applications centred on the concept of monitoring data as it passes through the system. By taking advantage of the available computing power, most applications operate in real time on live data streams. This paper has shown that a few rather simple concepts can be used to enhance multimedia applications. A general architecture has been presented, which categorises analysers based on their location on the network. Source, sink and network analysers have been defined and their choice is the result of trading off computational complexity, network congestion and application-specific needs. Analyser-enhanced services have been presented, where a range of audio and video processing units have been developed and employed.

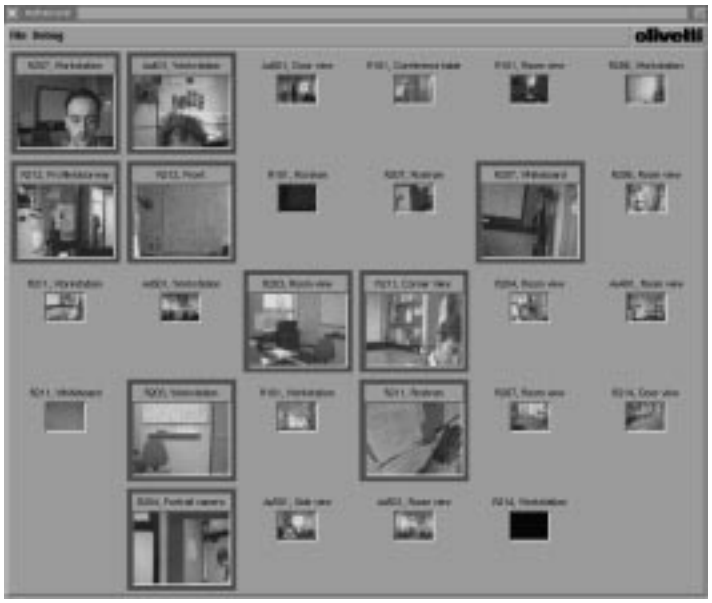

Figure 10: The security application

The analysers presented in this paper exploit features obtained from processing multimedia data. One area of interest for future work includes enabling analysers to learn patterns from observing user behaviour. Such analysers will be capable of acting autonomously on behalf of the user and be adaptive to new circumstances and requirements. Similar ideas have already been explored in related work on software agents [9].

Further work will investigate the possibility of using cameras and microphones in the context of location devices. By appropriately processing video and audio data, cameras and microphones can be used as sensors. Potential uses involve tracking and locating people as well as augmenting other existing location systems.

ACKNOWLEDGEMENTS

We wish to thank Tim Glauert, Martin Brown, Rob Walker and Frank Stajano of ORL, who have worked on the Medusa project and have contributed to this paper through discussions and by providing support software.

REFERENCES

1. P. Barham, M.Hayter, D.McAuley, and I.Pratt. *Devices on the Desk Area Network*. IEEE Journal on Selected Areas in Communication 13,4 (1995), 722-732.

2. A. Chaney, I. Wilson, and A. Hopper. *The design and implementation of a RAID-3 multimedia file server*. Fifth International Workshop on Network and Operating System Support for Digital Audio and Video (NOSSDAV'95, Durham NH, April 1995).

3. D. Clarke. *The medusa video brick: An ATM camera*. Technical Report 95.4, Olivetti Research Ltd, Cambridge (UK), 1995.

4. D. Greaves, D. McAuley, L. French, and E Hyden. *Protocol and interface for ATM LANs.* Technical Report 94.10, Olivetti Research Ltd, Cambridge (UK), 1994.

5. T. Heap and F. Samaria. *Real-time hand tracking and gesture recognition using smart snakes*. Technical Report 95.1, Olivetti Research Ltd, Cambridge (UK), 1995.

6. A Hopper. *The Medusa Applications Environment.* Proceedings of European Computer Support for Collaborative Working (Video version) (Stockholm, September 1995).

7. H.H. Houh, J.F. Adam, M. Ismert, C.J. Lindblad, and D.L. Tennenhouse. *The VuNet Desk Area Network: Architecture, Implementation, and Experience*. IEEE Journal on Selected Areas in Communications 13,4 (1995), 710-721.

8. A Jones and A. Hopper. *Handling audio and video streams in a distributed environment*. Proceedings of 14th ACM Symposium on Operating System, OSR 27,5 (December 1993).

9. P. Maes. *Intelligent software.* Scientific American (September 1995) 27,3, 84-86.

10. L.R. Rabiner. *A tutorial on Hidden Markov Models and selected applications in speech recognition*. Proceedings of the IEEE (1989) 77,2, 257-286.

11. F. Samaria. *Face Recognition using Hidden Markov Models.* PhD thesis, Cambridge University Engineering Dept., 1994.

12. F. Stajano. *Writing Tcl programs in the Medusa applications environment*. Proceedings of Tcl/Tk Workshop (New Orleans, June 1994).

13. A.M. Tekalp. *Digital video processing.* Prentice Hall, 1995.

14. R. Walker. *A speech recognition framework within the Medusa applications environment.* Cambridge University Computer Laboratory, Final Year Undergraduate project, 1995.

15. R. Want, A. Hopper, V. Falcao, and J.Gibbons. *The Active Badge Location System.* ACM Transactions on Information Systems, 10,1, 91-102 (January 1992).

16. S. Wray, T. Glauert, and A. Hopper. *The Medusa Applications Environment.* IEEE Multimedia, 1,4, 54-63 (1994).