Design and Implementation of a Software Intercom on LAN

Jonathan Gana KOLO, Umar Suleiman DAUDA, Gyet ALI-NOCK

Department of Electrical and Computer Engineering, Federal University of Technology, Minna, Nigeria
E-mails: jgkolo@gmail.com, usdauda@gmail.com

Abstract

In a developing economy like Nigeria financial resources are scarce commodity. Hence, cheaper means of accomplishing tasks even within the presence of scarce resources are always explored.

This paper presents the design and implementation of a software intercom on local area network (LAN). The design provides a solution that eliminates the use of expensive hard-wired intercoms and Private Automatic Branch Exchange (PABX) by making use of only LAN and full multimedia computer allowing network users the opportunity to still communicate audibly despite the meager resources.

The design enables the use of LAN that is mostly dedicated to data traffic only to be used as a cheaper means of communication using voice and video media, presented in real-time between peers by the use of software.

The software intercom designed provided network users with cost effective, easy and reliable multimedia communication over LAN, in essence giving the users a truly multimedia experience.

Keywords

Software; Intercom; Multimedia; Communication; LAN

Introduction

The need for information at the press of a button particularly in a developing economy cannot be overemphasized since telecommunication has long become the backbone in the day-to-day running of most businesses and organizations. Hence, the need to finding new solutions to make it cheap and affordable must be sought.

Currently, digital technology is the backbone of the entire information industry. As part of this, the transformation of audio information into digital signals is now a routine process that is incorporated into our telephone, digital networks, televisions and music equipments.

Voice communication traditionally has been carried over dedicated Telephone networks operated by Telecommunication service providers such as the Nigerian Telecommunications Limited (NITEL), Globacom Nigeria Limited (GLO-MOBILE), etc in Nigeria. These telephone networks have progressively evolved from the initial analog circuits to the current digital networks with bandwidth in excess of 1 Gbps. For reasons of varying bandwidth and networking requirements, different services are provided on separate networks. For example, telegraph networks, telex networks, telephone networks, Facsimile networks, Cable networks and Data networks support different services, as their names would suggest. These networks possessed characteristics that satisfied the peculiar requirements of the service they provided. For example, the voice network would support bandwidths of 64 Kbps for voice communication and would ensure Telco-grade voice communication with little jitter and echo cancellation [1][2]. Likewise, the cable networks would provide even higher bandwidth and improved quality of service (QoS) for video transmission. On the other hand, the data communication networks' bandwidth and QoS requirements are highly flexible. For most types of data communication applications, reliability is critical, which means that the delivery protocols would implement mechanisms for error checking, acknowledgment, re-transmissions and sequencing. On the other hand, for real-time applications such as voice communications, it would make little sense to retransmit a lost packet for play back at the receiving end, if it is out of sequence and is considerably delayed. Essentially, the main point to be noted is that these networks have been designed differently in terms of their underlying architecture and communication protocols.

With the immense growth of digital networks, which is the marriage of two technologies i.e. telecommunication technology and computer technology, Networks are being explored to the fullest. Ways, in which existing networks can be used optimally, with minimum additional cost, include facilities like video conferencing and mail serving incorporated into the network, saving cost, granting easier access to remote database and remote programs. Integrating these networks into a single integrated network, such that all services would use common facilities, presents a technological hurdle [3][4].

This project therefore is designed to provide a service that transmits voice and video over a Data network, as against having dedicated voice and cable networks meant for either voice or video communication only.

The advantages gained from having an all-purpose network as against having only dedicated networks include Cost Reduction, Simplification and Consolidation.

With the implementation of software-based intercom, the need for expensive PABX and telephone handsets would be eliminated since the only hardware requirements needed would be headsets and computers on network. Moreover, with the prevalence of Internet Protocol (IP) nodes and the abundant supply of IP based switches and routers, communication might not be limited to just LAN, but may include Wide Area Networks (WAN), thereby reduced the cost of making long distance calls. In addition, simplified installation and maintenance are major advantages the project will offer.

The major aim of this design therefore is to develop a LAN-based software intercom where anyone on the network can dial a peer that is logged onto the same network. This is an intercom without the PABX and expensive handsets. Another aim of the software intercom is to provide video peer-to-peer conferencing, where two or more clients can see each other in real-time using webcams or any other video capture devices connected to their computers.

Type of programming languages

Many high-level languages have been developed and designed; among these are BASIC (Beginner's All-purpose Symbolic Instruction Code), FORTRAN (FORmula TRANslation), PASCAL, COBOL (COmmon Business-Oriented Language), C-Language, C++ (Advanced version of C), JAVA, etc [5-7].

FORTRAN developed by International Business Machines Corporation (IBM) between 1954 and 1957 is used for scientific and engineering applications that require complex mathematical computation, but it is a text base programming language [8].

Dennis Richie at Bell laboratories developed C- programming language. The language is a very popular package among the computer user, it was first used to develop the UNIX operating system (a powerful multiuser, multitasking operating system. It is written in the C language and can be installed on virtually any computer). C++ is an extension of C, developed in the early 1980's at Bell laboratories. C++ provides a number of features that "spruce up" the C language with the capabilities for doing the so-called object-oriented programming (OOP). Many people believe that OOP can greatly improve the software development process [9].

JAVA was developed by SUN Micro system and released in 1995. JAVA is based on C and C++ and incorporates a number of features from other object-oriented language. JAVA includes extensive libraries for doing multimedia, networking, multi reading, graphics, database access and much more. This is why it is the programming language of choice for the software design [10].

Program Analysis

The Software Intercom has a graphical user interface (GUI) much like the various Internet phoning software found on the World Wide Web. The exception about this software is that it uses a Network Interface Card (NIC) to establish communication instead of a modem (as with most phoning software). When a caller dials the number of a peer, the sound file is activated which lets the call recipient know he is being called, and a pop-up dialog box appears on the screen interrupting whatever program is being run. The Implementation of this project is achieved with the JAVA Programming language.

JAVA Programming Language

JAVA is a high level programming language that was developed solely with the Internet in mind. Most web programs and handset used today are designed using the JAVA

programming language. It has unique attributes embedded into its design features, which make it the program of choice for most web application and software developers. Its embedded design feature include; OOP, Platform Independence, High Performance, Multi-Threading, and Dynamic linking. These entire features have made programming complex applications rather simple and straightforward. The design features are explained in detail in [11-14].

Program Design

The major challenge in the designing and running of this program was capturing the analog sound and video images from the microphone and web cam, buffering it and transmitting it seamlessly to the specified destination in real-time mode. On the other side the recipient should be able to hear and reply in a two-way communication (full duplex mode). The fig. 1 below shows the software startup.

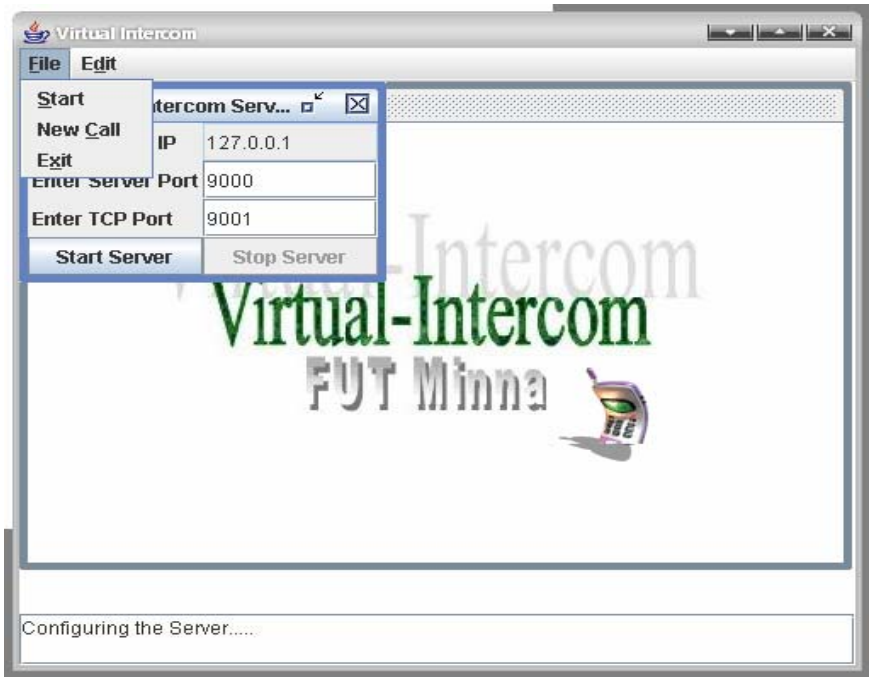

Figure1. Showing the program startup with its Server page

When a caller wants to make a call to another client on the network, he runs the program by clicking an icon on his desktop, then from the program's edit menu he chooses

"configure server" option which configures his local server to listen for any incoming calls from other callers on the network. He then starts his Virtual intercom by choosing the "Start" option from the file menu. Finally, the caller loads a window called "New Call" from the file menu. This window is shown in Fig. 2.

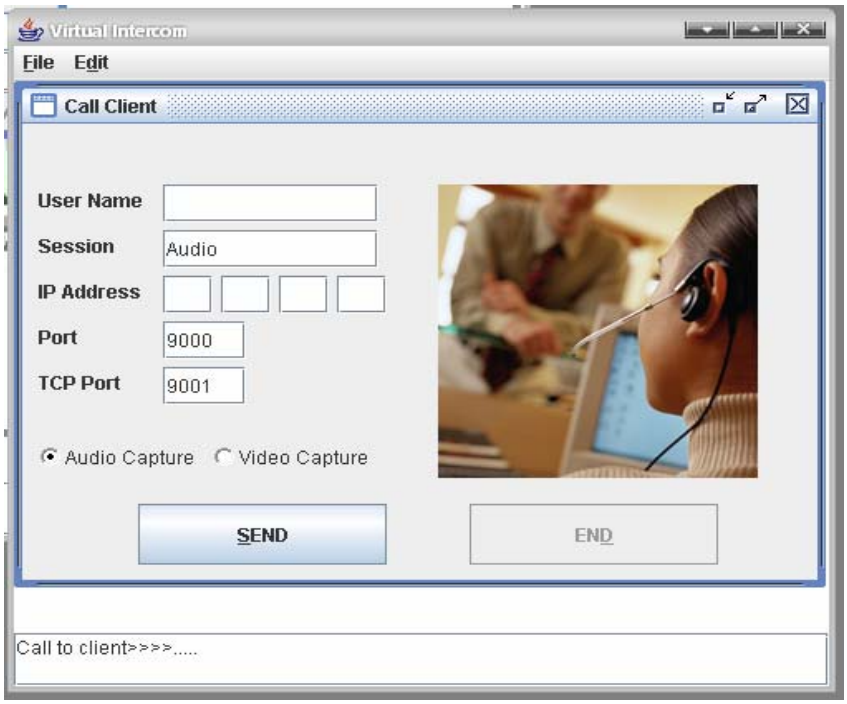

Figure2. Showing the Call Client

The window or internal frame contains several parameters the caller needs to use to establish a connection with another on the network i.e. a text-field displaying the caller's identity and other fields to input the recipient systems IP addresses, and also a choice of what type of media is to be captured, i.e. audio or video. He/She then makes the call by clicking on the send button and the specified recipient is prompted about an incoming call. As long as the call parameters are correct and the other person's software is running, then secure link between the two peers will be established.

Installation

The software package designed using JAVA must be installed on each system the application will run on. The program automatically acquires the IP address of the system on which it is installed, so that its local server can be easily configured without much problem.

The program uses a JAVA Runtime Environment (JRE 1.5.0) to execute and also JAVA Media Framework (JMF), which is an Application Package Interface (API) that does not come with the standard JAVA development toolkit. All other IP addresses of the other systems on the network would have to be obtained manually. These are important because these addresses are used in place of phone numbers to establish the connection between peers.

Systems Requirement and Implementation

Software Requirements

The aim is to enable the software to run on most computer platform (operating system). To do so would require some software: application packages and API must be installed basically on every computer system to enable the software intercom to run effectively.

Operating System (OS)

An OS manages the resources of the computer system i.e. hardware and software resources. The software runs on any of the major operating systems available in the market or in the information technology (IT) world. These OS are **WINDOWS 98, 2000, NT, XP, AND VISTA**; others are **LINUX,** and **UNIX.**

Runtime Environments and API'S

Since this program was written in JAVA, it needs a runtime environment to interface it with the operating system of choice. It uses a **JRE 1.5.0** (Java Runtime Environment version 1.5.0). Another very important API needed to run this program is **JMF 2.0** (Java Media Frameworks 2.0). Its importance and purpose is highlighted below:

JMF 2.0 is an application-programming interface for incorporating time-based media into JAVA application and applets. This API supports media capture and addresses the needs of application developers who want additional control over media processing and rendering. It also provides a plug-in architecture that provides direct access to media data and enables JMF to be more customized and extended.

Hardware Requirements

A computer with full multimedia capabilities is the basic requirement for running the program. There should be installed a good full duplex sound card, with working speakers and a microphone. Since the software is also capable of transmitting videos, a television (TV) card could also be installed too. This forms the interface needed for connecting a video camera to the computer. A webcam can be used instead of a video camera that requires a TV card, primarily because the webcam can be connected to the computers universal serial bus (USB) port with no need for extra expensive hardware. The only draw back to using a webcam is that the quality of pictures is greatly diminished.

Local Area Network (LAN)

Today local area networking is a shared access technology. This means that all of the devices attached to the LAN share a single communication medium, usually a coaxial, twisted pair, fiber optic cable, or wireless medium in other cases. Several computers are connected to a single cable that serves as the communications medium for all of them.

Installing Network Interface Card (NIC) in each computer and connecting it to the network cable establish physical connection to the network. Once the physical connection is in place, it is up to the network software to manage communication between stations on the network. The LAN should be of quality speed [15][16].

Program Implementation

The system works with both point-to-point and point-to- multipoint configuration with full duplex transmission for voice and video conferencing. This means that the software works both as client and server software. The purpose of the server is to listen in on any communication in the network that is addressed to the IP address of the computer it is resident on. It then relays the message to the client addressed which in this case is the computer it is running on. This is shown in Fig.3.

When the program is launched the software is configured to set the machine to receive any incoming calls from another machine on the network. By default the server is initialized through Transmission Control Protocol/Internet Protocol (TCP/IP) socket-socket communication on a selected port (9001). When a call is being setup, the transmitting station is required to provide user identification that is transmitted along with its IP address to the

receiving end. This enables the receiving end to know what machine on the network is trying to establish a connection.

The transmitting system then sets up a socket communication with the receiving station. During this call setup phase, the transmitting and the receiving station agree on transmission parameters like the Real-Time Transport Protocol (RTP) session port, the audio quality signal (this particular implementation uses the base sampling quality on the system, specifically using the following audio format; LINEAR, 8000.0Hz, 16-bit, Mono, LittleEndian, signed) and the capture device used is Java Sound Audio Capture locator = java sound;//44100 which represents the system microphone. Other capturing system include the system dependent Direct Sound Capture (locator = dsound ://).

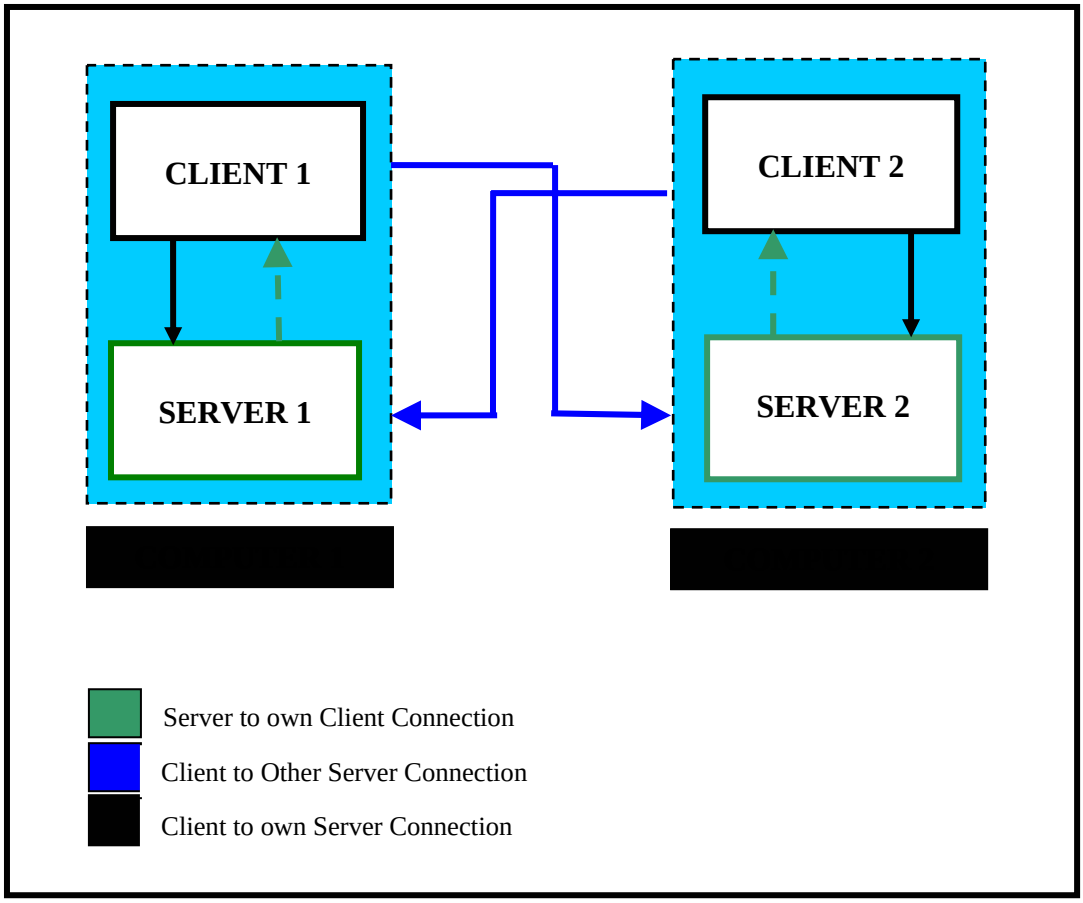

Figure 3. A Pictorial view of Peer-Peer communication of two Computers running the software

After the socket communication has agreed on the parameters to be used for the communication, an RTP session is started based on the parameters to be used for the

communication. This RTP session could be started based on the parameters, and real-time audio could be captured and transmitted between the two systems in full duplex point-to-point configuration.

Program Segments

Basically, there are four modules or segments:

1. Capturing of media data from the input device.
2. Encoding of the captured audio data.
3. Transmitting the captured and encoded data.
4. Decoding and rendering the data stream at the destination.

(i) Capturing Media Data

To capture media data:

Locate the capture device to be used by querying the CaptureDeviceManager.

- Get a CaptureDeviceInfo object for the device
- Get a MediaLocator from the CaptureDeviceInfo object and use it to create a DataSource.
- Create a Player or Processor using the DataSource.
- Start the player or processor to begin the capture process.

(ii) Encoding Captured Audio Data

The processor can be configured to transcode captured media data before presenting, transmitting, or storing the data. To encode captured audio data in the IMA4 format before saving it to a file:

1. Get the MediaLocator for the capture device and construct a processor.
2. Call configure on the processor.
3. Once the processor is in the Configured state, call getTrackControls.
4. Call setFormat on each track until one that can be converted to IMA4.
5. Realize the processor and use its output DataSource to construct a DataSink to write the data to a file.

(iii) Transmitting The Media Stream

The basic process for transmitting RTP data with the session manager is:

1. Create a JMF processor and set each track format to an RTP-specific format.
2. Retrieve the output DataSource from the processor.

3. Call createSendStream on a previously created and initialized SessionManager, passing in the DataSource and a stream index. The session manager creates a SendStream for the specified SourceStream.
4. Start the session manager by calling SessionManager start Session.
5. Control the transmission through the SendStream methods. A Send Stream Listener can be registered to listen to events on the Send Stream.

(iv) Decoding and Rending The Media Stream

1. Set up the RTP session
    a. Create a **SessionManager.** For example, construct an instance of **com.Sun.media.rtp.RTPSessionMgr**. (RTPSessionMgr is an implementation of SessionManager provided with the JMF reference implementation.)
    b. Call **RTPSessionMgr.addAVStreamListener** to register as a listener
    c. Initialize the RTP session by calling the **RTPSessionMgrinitSession**.
    d. Start the RTP session by calling the **RTPSessionMgrstartSession**.

2. In **AVStreamListener** update method, watch for **NewReciveStreamEvent**, which indicates that a new data stream has been detected.

3. When a **NewRecieveStreamEvent** is detected retrieve and **RecieveStream** from the NewRecieveStreamEvent by calling **getRecieveStream.**

4. Receive the RTP **DataSource** from the **ReceiveStream** by calling **getDataSource**. This is a **PushBufferDataSource** with an RTP-specific Format. For example, the encoding for a DVI audio player will be DVI_RTP.

5. Pass the Data Source to **manager.createplayer** to construct a player. For the player to be successfully constructed, the necessary plug-in for decoding and depacketizing the RTP-formatted data must be available. The basic files used are:

    1. AVRequestListener.java
    2. AVStreamListener.java
    3. AVStreamTransmitter.java
    4. AVserver.java
    5. Vintercom.class

Results and Discussions

The designed software intercom was installed on 10 systems on a particular LAN running at the data rate of 100Mbps. The 10 peers on the network were able to establish audio and or video communication with each other successfully via the LAN. Hence, the designed software intercom worked successfully.

With the implementation of the software-based intercom, expensive PABX and telephone handsets were eliminated since the only hardware requirements needed were headsets and full multimedia computers on the network.

Ten Files were involved in the writing of the software program; all forming separate modules with specific functions, taking advantage of java's object oriented programming properties. For the purpose of documentation we have excluded all files that were used to create the GUI (graphical user interface) and included only those that form the core of the program itself. These files are AVRequestListener.java, AVStreamListener.java, AVStreamTransmitter.java, AVServer.java, RemoveUserEntry.java, callPanel.java, startPanel.java, execPanel.java, SoundPlayer.java, Vintercom.java.

Conclusions

In this paper, we presented the design and implementation of a software intercom on LAN using JAVA programming language. Companies with good quality network installation can enhance the functionality of their network by using this software intercom. This eliminates the need for expensive PABX. The installation of the software intercom is simple and easy. The protocols involved in the program design do not reserve bandwidth, so this will not slow the network down. More so, it shows how standard system can be assembled to support a multimedia application such as audio and video conferencing over networks. The trend toward telecommuting and virtual office has created both near-ideal condition and a legitimate business need for audio and video conferencing.

The architecture of this software has been designed and tested to provide network users with cost effective, easy, reliable multimedia communication over a network.

References

1. Aaron U., *Phones for all.* A publication on the past, present and future of Nigerian Telecommunications industry, 2003, pp. 7-19, 31-34, 69.
2. Moses D., *Telecommunication Development situation in Democratic Nigeria*. A paper presented at the world Telecommunication Development conference, Istanbul, 18-27 March, 2002.
3. Debashish M., *Network Convergence and Voice over IP*, TATA Consultancy Services, 2001.
4. Quitum Technologies Inc, *Risk and Rewards: 2000 Strategies for Migrating Corporate Voice Traffic to data Network*, Quitum Technologies Inc. 14 Christopher ways Eatontown, NJ 07724 [online]. Available at www.quitum.com
5. Kogge, Peter M., *Computer Program.* Microsoft Student 2008 [DVD]. Redmond, WA: Microsoft Corporation, 2007.
6. *Pascal (computer).* Microsoft Student 2008 [DVD]. Redmond, WA: Microsoft Corporation, 2007.
7. *Java (computer).* Microsoft Student 2008 [DVD]. Redmond, WA: Microsoft Corporation, 2007.
8. *Fortran.* Microsoft Student 2008 [DVD]. Redmond, WA: Microsoft Corporation, 2007.
9. *UNIX*. Microsoft Student 2008 [DVD]. Redmond, WA: Microsoft Corporation, 2007.
10. Hassan A. B., Abolarin M. S., Jimoh O. H., *The Application of Visual Basic Computer Programming Language to Simulate Numerical Iterations*, Leonardo Journal of Sciences, Issue 9, July-December 2006, p. 125-136 [online]. Available at http://www.ljs.academicdirect.org/A09/125_136.pdf
11. Elliotte R. H., *Java Network Programming*, 3rd Edition, Published by O'Reilly Media, Inc., 1005 Graven stein Highway North, Sebastopol, CA 95472 [online]. Available at http://safari.oreilly.com
12. Deitel and Deitel, *Java: How to Program*, 4th Edition, Published by Prentice Hall, 2002.

13. *JMF 2.0 API Specification,* from java.sun.com [online]. Available at: http://java.sun.com/products/java-media/jmf/2.1/specdownload.html

14. Budi Kurniawan, *Program multimedia with JMF, Part 1*, Apr. 06, 2001 [online]. Available at http://www.javaworld.com/jw-04-2001/jw-0406-jmf1.html

15. Midkiff, Scott F., *Network (computer science).* Microsoft Student 2008 [DVD]. Redmond, WA: Microsoft Corporation, 2007.

16. Midkiff, Scott F., *Local Area Network.* Microsoft Student 2008 [DVD]. Redmond, WA: Microsoft Corporation, 2007.