Information Sharing Using a Client Server Prototype: M to VB via TCP/IP

Robert D. Andrews
VA Information System Center
Salt Lake City, Utah

ABSTRACT

This paper describes a prototype application that shares information between a central M database and a non-M based Windows application over a TCP/IP network. The Department of Veterans Affairs' (VA) Forum account contains a wealth of information on Decentralized Hospital Computer Program (DHCP) applications, users, sites, and requests for customer service. This information can be made available over a wide area network (WAN) to remote workstations. Two strategies allow for accessing the information: periodically offloading information to the workstation, and real-time transfer of data between the M server application and the non-M client workstation using TCP/IP. This prototype provides an easy-to-use interface for geographically distant workgroups and explores some interesting methods of managing distributed data in a client server environment.

INTRODUCTION

One of the major reasons for the success of the VA's DHCP is the communication among sites via network mail to Forum (a central Email account). Forum also has national databases of site profiles, users, package information, NOIS (National On-line Information Sharing), and software patches. Information sharing allows geographically remote users to solve problems, ask questions, give feedback, and benefit from the wide wealth of talent throughout the VA. The intent of this prototype is to allow even easier access for finding current information and logging requests for service by incorporating the data in a more intuitive, windows-based application.

THE WORKSTATION

The client application is a Visual Basic (VB) program running under Windows for Workgroups (WFWG). The program installs as a separate window application or runs from a terminal emulator script. The VB program uses data from the M server. These data are either periodically extracted and downloaded from the host system or made available directly using a network connection. When the application is running on a PC without a network connection, static information is available. When connected to a network and set up to use TCP/IP, the application takes advantage of real time data transfers with the server.

Figures 1 and 2 show examples of the workstation application.

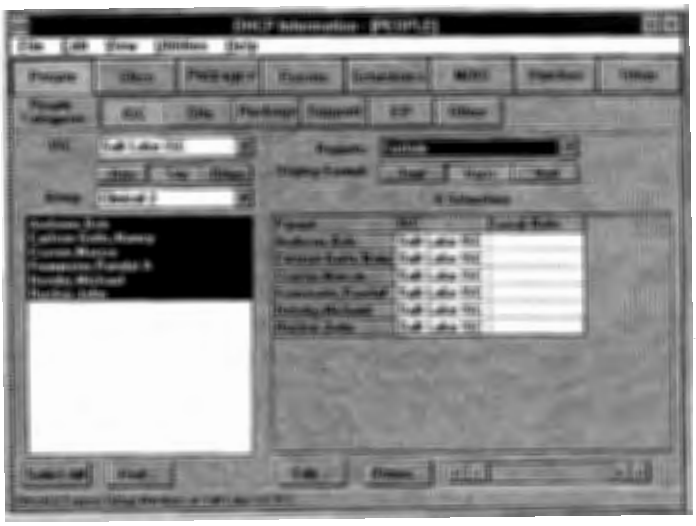

Figure 1. Workstation showing information on people.

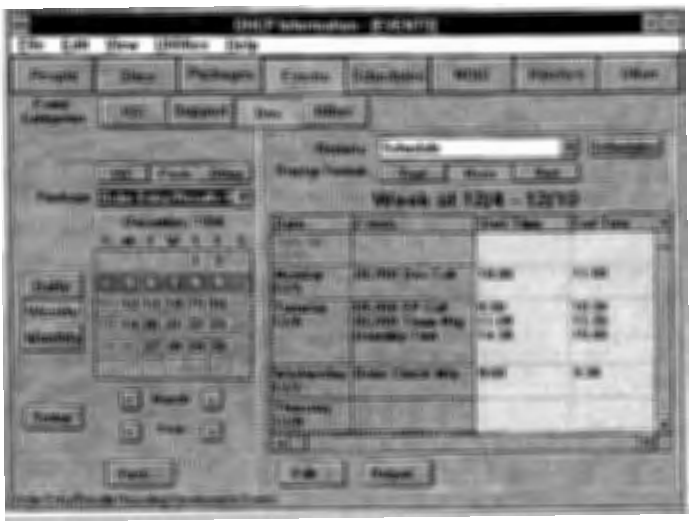

Figure 2. Workstation showing events..

The program presents a top row of buttons representing information for different subjects such as People, Sites, Packages, Events and Patches. Clicking on a button changes the subject. Figure 1 shows information available when a user clicks on the People button. A series of other buttons filters these subjects further. For example, Figure 2 shows information filtered to display only information on development events concerning the Order Entry/Result Reporting package for the second week in December.

Information displays in various formats. Ad hoc reports allow custom displays. The VB program allows reports to be defined using Structured Query Language (SQL) statements that can be stored in a VB database file. SQL statements are also used within the workstation program to filter the selections of subjects. SQL statements are only used by the user if permanent reports are created; otherwise, information is simply displayed by clicking buttons.

Transferring information to other Windows applications is easy. For example, a user creates a phone list by selecting a group of people and transfers the data to a table in a word processing (or spreadsheet) document by simply choosing a table format for the output. The word processor or spreadsheet must support object linking and embedding (OLE). The VB program makes use of OLE automation by using the application's macro language as a new object. For example, the VB command CreateObject("Word.Basic") allows the Microsoft Word for Windows macro language to be used within the VB program to automatically create and format a table within a Word document. This functionality takes advantage of using the client for custom processing.

Read-Only Information

The subjects for People, Sites, Packages, Patches, and Other are similar to a rolodex for finding information such as addresses and phone numbers. This information is kept up-to-date through data entry on Forum by individuals using their normal access to the M database. These data are generally static and are only supplied to the workstation by periodic downloads. Downloads are workstation configurable and can be as often as once a day. This information is read-only on the workstation and does not require a TCP/IP network connection.

Time Oriented Information

The Event's calendar allows reviewing dates for various groups of users. Meetings, conference calls, training, and other events can be entered into an M database. This information is periodically downloaded to the workstation similar to other read-only data. These events are displayed in the user's time zone. This information is helpful in showing conflicts of events and in working on the same projects across geographic boundaries.

The Schedule application shows the availability of support specialists for helping with a problem. The information contains the user's normal working hours, schedules, or events. Events are imported from the workstation calendar application or by using Schedule+ on the PC. Schedule+ is a standard application with WFWG that has an application program interface (API) for sharing information with VB [1, 2]. Viewing the availability of others allows specialists from different locations to work in groups on common applications. Scheduling requires a TCP/IP connection, in order to read and write to the M database.

NOIS

On Forum, NOIS is an M application used to log requests for service from VA sites (calls regarding software and hardware problems). Logging calls, reviewing calls, and searching for calls use conventional M programs on Forum. NOIS is designed to set data as transactions and to retrieve data using search definitions (relatively simple query statements). NOIS information is heavily indexed to allow for fast retrieval. NOIS is ideal for acting as a server to client applications.

On the workstation, NOIS will be fully operational as a client to the host application on Forum. The workstation even has features not found on Forum. Summaries of related information will be available that are relevant to a given problem. For example, while logging a call regarding an automated instrument in the Laboratory package, you select a summary option. The NOIS workstation responds by providing the other sites that have this instrument, the names and phone numbers of application coordinators at those sites, Information Systems Center (ISC) support staff available (at this very moment), and any related patches or previous NOIS entries. Hopefully, a solution is already available. NOIS requires a TCP/IP connection for interactive use; otherwise it is disabled or restricted to read-only access of downloaded data.

CLIENT/SERVER/NETWORK MODEL

This application makes use of the WFWG 3.11 winsocket that allows TCP/IP to be available to Visual Basic and an M system that also supports TCP/IP. Information is made available by two methods: 1) periodic offloads of static data from the server to the client for read-only access, and 2) interactive, read/write access between the client and server.

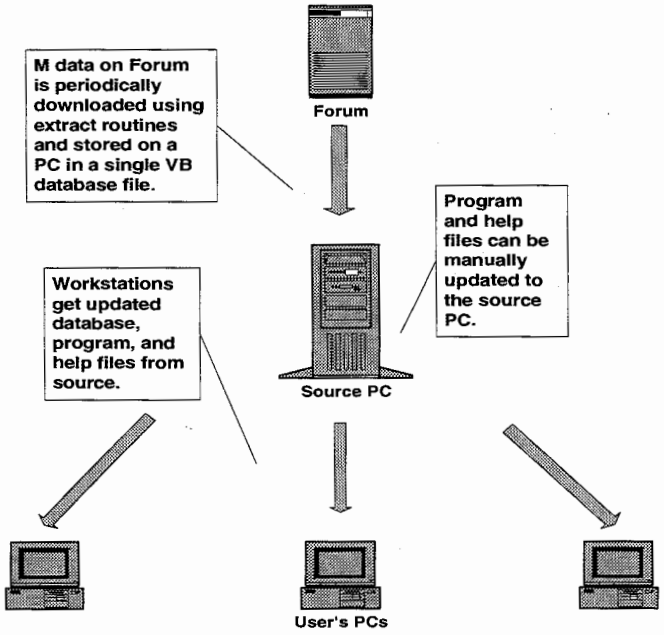

Figure 3. Download distribution uses periodic samples of M data copied to workstations.

Distributing Downloaded Information

The process of downloading information from Forum to workstations is shown in Figure 3. Downloading applies to data for the subjects: People, Site, Package, Events, and Patches. A nightly task runs an M routine that transfers extracted information from Forum to text files on a PC using TCP/IP through a VB program. This process can also be done manually by "capturing" an output of the M routine using a terminal emulator script. The text files are then processed on the PC by a VB program into a single VB database file. This database file resides in a shared directory on the ISC's WAN. Access to this "source" directory (read-only) is available to anyone at the ISCs using WFWG. When the workstation application is started, it automatically checks the source database to see if the file's creation date is more recent than the active database on the workstation. If the database is out of date, the source database is copied in as the active database on the workstation.

The technique of copying more recent files also applies to help files and the workstation program itself. The application, database, and documentation are maintained automatically (see Figure 4). This provides a means of "auto-maintenance" and allows enhancing the application without reinstalling files manually. If the application cannot access the network source directory (examples: the network is not used, using a portable

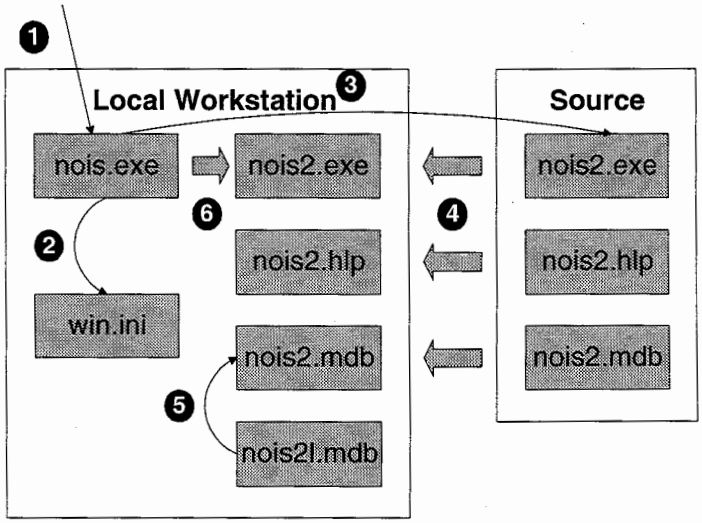

Figure 4. Auto-maintenance allows easy updating of patches and enhancements.

Note: Although transparent to the user, the workstation checks whether or not to update before starting.

1) Nois.exe is a small 'starter' program. Nois.exe is run from an icon. It checks for proper configuration and starts the workstation application (nois2.exe).
2) Nois.exe checks the configuration in the win.ini file.
3) Source files are checked to compare their date/time of creation to the same files on the local system.
4) Three types of files may be copied to the local system: the program, help file, and database.
5) When the workstation database is copied over, all local data is lost. To restore this, the update process automatically restores local data (stored redundantly in nois2l.mdb) to the active database (nois2.mdb).
   The nois.exe routine starts the real workstation application (nois2.exe). After the workstation application has been started, the nois.exe routine ends.

computer, or, the network is down), the update check is ignored and the application is started. Updates are also configurable on the workstation (see Figure 5). Note in the figure that a universal path is used (the \\ syntax). This allows file copies without having to directly make a virtual drive connection on the WFWG network.

Data on the client can also be stored locally without updating to the host. Data (such as notes on sites) can be stored in the active database. Local edits are stored in a separate file on the workstation. These edits are automatically added back to the active database whenever the database is copied over. This

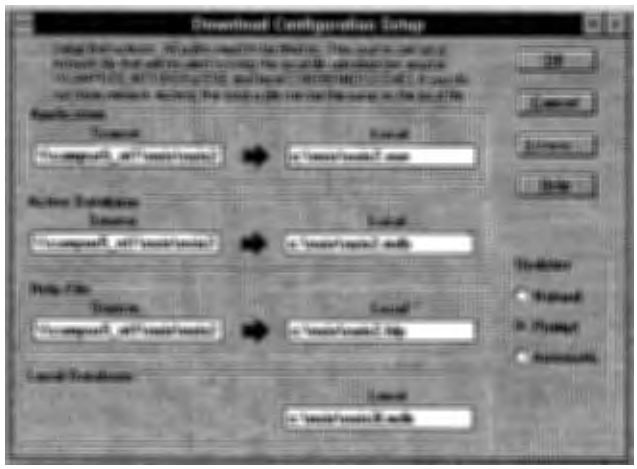

Figure 5. Updates are configurable on the workstation.

allows keeping "private" information that is not updated to the host. Actually, these edits are only private to the workstation and not the user, so these types of edits are only useful for those using a specific workstation.

Network Interaction of Client and Server

The workstation application communicates interactively with the M host using TCP/IP. TCP/IP allows connections over a WAN so that geographically distant clients can work from the same host. A dynamic link library (DLL) utility — the winsocket DLL — allows Windows to use this network protocol [3, 4]. M vendors also support TCP/IP by designating a special IO device number.

The VB and M communication utilities were written by Kevin Meldrum at the Salt Lake City ISC for use with the Order Entry/Result Reporting application. The utilities allow an M job to open and use a device dedicated to TCP/IP in order to read and write to the winsock.dll. The VB utilities allow for connecting and disconnecting to the winsock.dll and making calls to send and receive data to the M application (see Figure 6). Using these utilities, it becomes a simple task to write routines for each end (M and VB) to export and import data.

The workstation application uses TCP/IP for schedules and NOIS interactions. For example, to display the availability of users and their current status (in/out/busy), the users are selected on the Schedules form. A message is sent from the client to the server requesting the current statuses. The server runs an M routine and returns the data.

CURRENT STATUS

This model is being tested by downloading data from Forum using a terminal emulator. The download operation works well but it is not completely automated. The interactive process that uses TCP/IP is being tested from an account at the Salt Lake City ISC where NOIS and Site Tracking packages are installed similar to those on Forum. Once it is well tested, the server will be switched to Forum.

A representative sample of data extracted from Forum requires about 3 hours of processing to captured text files using a terminal emulator at 9600 baud rate. It's estimated this would be less than 1 hour when using TCP/IP. The text files can be processed to a VB database in about 5 minutes. The database size is approximately 8 Mbytes. Copying the database from the source directory to the workstation takes about 20 seconds (using a 66 MHz 486 with 16 Mbytes of memory).

At this time, only the scheduling application uses interactive TCP/IP. It has been tested with fifteen simultaneous users updating and redisplaying scheduling data. An impressive demonstration of this technology is made when people at different ISCs (ex. Salt Lake City and Albany) simultaneously change their availability status on the workstation application

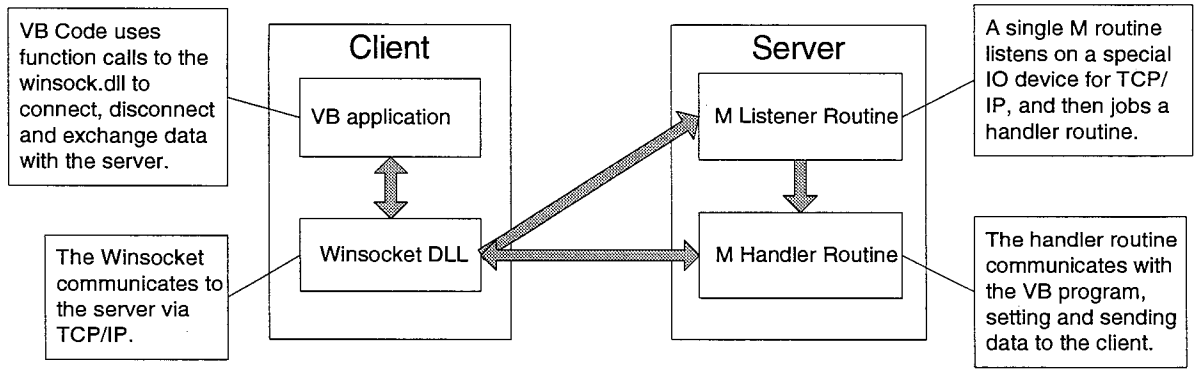

Figure 6. Client communicating with server via TCP/IP.

(including a message with text and background colors) and each user sees the corresponding changes displayed on their own PC. It is also impressive to observe the changes to the M global in the Salt Lake account while these interactions are going on. It works!

NOIS will be fully developed to be used interactively. It will be very interesting to compare the search speeds of a large M database via TCP/IP to an equivalent local VB database.

Usability tests have been conducted on the program, and the application has been improved to be more intuitive to users. For example, new users often found it difficult to filter the selections of the various information. The program was revised to use better labeling, a status bar message was added, and focus was set to the most likely selection. When a new set of users tested the changed interface, their completion times were shorter on the same tasks.

CONCLUSIONS

A client server system allows sophisticated, easy to use, easy to learn, and fast user interface for multiple users from a central pool of information. The server focuses on performing a reduced set of tasks (retrieving and updating data) while the processing of the information is offloaded to the clients.

The strength of this model is that it makes use of each client server component (data layer, communication layer, and user layer). The M language is an excellent tool for managing data and communicating information between multi-user environments. The standard network protocol, TCP/IP, allows connections across a WAN. Windows and VB are great tools in providing a user interface. Since each component is independent, future enhancements and replacements of a component are less painful to incorporate.

Key features of this prototype include:

- **Stand Alone** — A substantial amount of information is accessible, even without network access.
- **WAN** — Sharing up-to-the-minute information with others around the country is no problem.
- **Daily Distribution** — Downloaded information is available daily with little impact on the server.
- **Auto-maintenance** — Patches and enhancements are automatic.
- **VB Database** — Offloaded information is inexpensive since the copied VB database does not require multi-user licensing for distribution or use.
- **Special VB Features** — OLE, SQL, and direct calls to the Windows API produce sophisticated functionality.
- **Point and Click** — The workstation's forms only require single clicks to get information. Advanced features requiring text entry are always on secondary dialogs (ex. a search for patches containing a word or phrase).
- **Time Zones** — Schedules and events are viewed with the appropriate time zone.
- **Private Information** — Local entries can be added.
- **NOIS on Forum** — NOIS's indexing and query capability produces an ideal multi-user server.
- **Merged Downloads** — Downloads could be merged from multiple systems (ex. Albany ISC's SAGG data showing sites global growth).
- **Database Comparisons** — Large databases in M and VB can now be benchmarked and compared.
- **Implementation Friendly** — By implementing this prototype at ISCs, it can be fine-tuned before distribution to Information Resource Management Services at medical centers, and as networking permits, be distributed to DHCP application users.
- **Help Desk on Your Desk** — The real goal is to have answers available.

This prototype explores some of the tradeoffs of offloading information. It also positions itself to take advantage of new technology that might be easily incorporated (such as telephony API). Potentially, it has the ability to allow remote users to work together more easily. Sharing problems and solutions should reduce problems and allow users to do their jobs better.

ADDITIONAL INFORMATION

[1] Microsoft Corporation, "The Schedule+ Access Libraries for Visual Basic" (1994) in *Office Developer's Kit* (CD).

[2] R. Jennings, "Interacting with Microsoft Mail, Schedule+, and Telephones," in *Database Developer's Guide with Visual Basic 3* (Indianapolis, IN: Sams Publishing, 1994).

[3] Hewlett-Packard Company, "Programmer's Reference, Sockets for Microsoft TCP/IP" (1991) in *Microsoft Development Library* (CD 10).

[4] D. Treadwell, "Developing Transport-independent Applications Using the Windows Sockets Interface," Tech*Ed 1994 Conference in *Microsoft Development Library* (CD 10).