INTEROPERABILITY OF LARGE MUMPS SYSTEMS WITH OUTSIDE SYSTEMS, DEVICES, SOFTWARE AND DATA IN A DISTRIBUTED ENVIRONMENT

Ruth E. Dayhoff, M.D. and Peter Kuzmak
Washington Information Systems Center
8403 Colesville Road, Suite 200
Silver Spring, MD 20910
(301) 427-3700

Abstract

The Dept. of Veterans Affairs (VA) Medical Imaging Project has used a number of mechanisms to allow MUMPS to interoperate with non-MUMPS systems, processes, and data. These include the use of remote procedure call mechanisms, shared mass storage or memory access, and direct process-to-process communication schemes. For MUMPS systems to compete with industry developments, they must be able to operate in standardized interface environments in order to make use of economical off-the-shelf reusable software. In addition to its other strengths, MUMPS is outstanding as a command and control language that can glue together software modules to create large systems made up of many heterogeneous components. The ability to control non-MUMPS software provides new capabilities to MUMPS systems.

Introduction

The Dept. of Veterans Affairs has developed its own hospital information system (HIS) called the Decentralized Hospital Computer Program (DHCP) which runs in virtually all of its 171 hospitals. This is a very complex system, encompassing many software modules and including thousands of menu options. In the past, most of its modules have involved the acquistion, storage and retrieval of ASCII data. Over the past three years, this system has been extended to allow the handling of non-text data such as medical images, scanned documents, and electrocardiogram (ECG) wave forms. This has required the development of a number of interfaces for the acquisition, handling, storage, and control of outside data and processes. These new data types and functions have been integrated in such a way as to be invisible to the user. In supporting user decision making, the system must present the user with related data in a way that allows the person to think about it best, not in a way that reflects the internal system's storage mechanisms.

Recently, the computer software industry has recognized the need to deliver general software modules that can be integrated by many different software packages written in a variety of languages. Such reusable modules reduce overall costs in several ways:

- Allowing testing to occur in a centralized manner, with increased reliability as a benefit to all users.
- Saving software developers the effort of rebuilding functionality themselves in their environment
- Allowing modules to be used as standalone packages or called by other applications

System environments have provided support for this concept by standardizing the interface environment for such software modules, by providing remote procedure call mechanisms, shared mass storage or memory access, or by direct process-to-process communication schemes. For example, Microsoft Windows 3.1 has provided Dynamic Data Exchange (DDE), Dynamic Linked Libraries (DLL), and Object Linking and Embedding (OLE) as "standard" mechanisms by which processes can communicate and control each other. UNIX provides Remote Procedure Calls (RPC) for connected systems. Networked file servers containing mass storage devices as well as caching memory offer a simple alternative for intersystem communication and control. In fact, network operating systems such as Netware are moving to include more services directly to users or to programmers through application program interfaces [1]. The DHCP Imaging Project is making use of a number of these mechanisms to provide device control, and integration of data and processes.

For MUMPS systems to compete with industry developments, they must be able to operate in

standardized interface environments in order to make use of economical off-the-shelf reusable software. The strategy for making MUMPS system interoperable with these environments should be to use existing facilities wherever they are available and not try to recode all applications in MUMPS. In addition to its other strengths, MUMPS is outstanding as a command and control language that can glue together large industrial-strength modules to create large systems made up of many heterogeneous components.

Data Integration Needs

The DHCP Imaging System uses workstations networked to the DHCP hospital information system to integrate a variety of types of medical data with patient's online records. These include high resolution true color or black-and-white images, slow-motion or compressed video images, scanned document images, digitized waveform data, and audio data. Software to control these various data types may be written in a variety of languages.

The VA's hospital information system needs to integrate data from a number of sources, including:

- Completely separate systems, such as radiology Picture Archiving and Communication Systems (PACS) or Electrocardiograph Acquisition Systems (ECG)
- Workstation-based scanning devices such as Laser Xray Scanners that run with vendor supplied software
- Workstation-based scanning devices supported by software running as a separate process on the workstation, such as document scanners
- Modem or fax mediated data communication

A variety of data communication solutions have been implemented depending on the type of external software or system involved. Most of these solutions involve the reading and writing of files on file servers, shared disk or shared memory. In some cases, special processes may be needed to display the data.

Approaches to Integration

The VA's MUMPS-based DHCP Imaging Project has used a number of approaches to achieve data communication and execution of non-MUMPS software (Figure 1):

(a) *Communicating messages between separate systems as files written to and read from network file servers.* This method was used to pass ACR-NEMA standard messages bidirectionally to a radiology PACS system [2,3]. These messages contain either text or images or both. This same technique has been proposed by the VA for transfer of ECG waveform data, although a standard message or data format does not exist. This method is also used to communicate with a PC-controlled laser xray scanner using a mutually defined communication file format. The xray scanner runs a monitor program, which watches for message files on the network server and responds to them by scanning the mounted xray and placing the resulting file on the server. The program then returns to monitor mode while the HIS software handles the scanned file and user interaction (Figure 1A).

(b) *Communicating messages via RAM disk files (or pipes) to a local process server, which in turn communicates with device control software.* This mechanism has been used to communicate with document scanning software running as a Microsoft Windows process. The process server receives a message to invoke a suppported function; it opens a window and begins the selected function, receives user interaction, and closes the window (Figure 1B).

(c) *Communicating messages via Dynamic Data Exchange (DDE) supported by Microsoft Windows Version 3.1.* A prototype implementation has been developed that allows the hospital information system to control a scanning package based on Microsoft Windows via DDE messages [4]. A wide variety of scanner control functions are available through DDE (Figure 1C).

(d) *Executing processed directly via calls to external functions.* We have made extensive use of proprietary ZCALL interfaces to communicate with Terminate and Stay Resident (TSR) modules in MS-DOS. Our DHCP Imaging System has twelve function calls that perform basic imaging operations. The MUMPS Development Committee is in the process of passing the External Call proposal, which will standardize the external call interface from MUMPS to other languages. Several vendors have already implemented

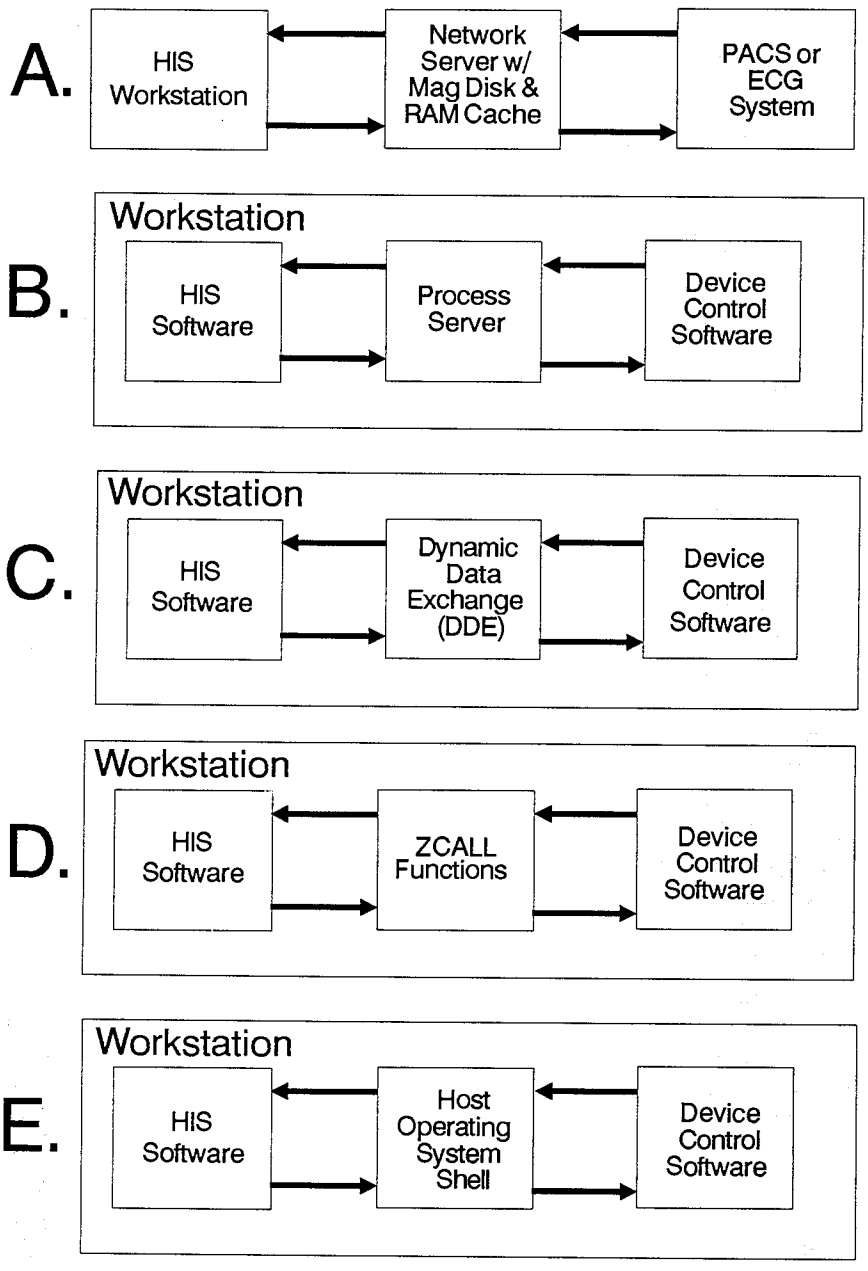

Figure 1: Five approaches to achieve data communication and execution of non-MUMPS software

this external call interface (Figure 1D).

(e) *Executing processes through proprietary host operating system interfaces.* The ability to invoke the host operating system shell from inside MUMPS is a very useful feature of some versions of MUMPS. This feature allows any command line script to be run from within MUMPS and greatly extends the capabilities of the application. We have tested this mechanism for some image functions and for transmission of multimedia mail messages between sites [5] (Figure 1E).

Object-Oriented Data Structures to Support Integration

An object-oriented file structure is used to support data access, interrelationships of data, and integration of processes [6,7]. This file structure is managed by the VA's database management system, VA FileMan [8]. A file entry exists for each data instance, for example, an image. That entry stores a reference to the operating system file. In addition, each data entry includes a pointer to its data type and pointers to its network location. A data instance can have multiple network locations, for example, when an image is stored permanently on an optical jukebox but is also temporarily cached on a magnetic server. Remotely located data can be accessible transparently, as though it were on a local drive, or it can be reached via messages passed through a gateway or broker service. It may be reached by either method at any given time, with the software selecting the priority for access (Figure 2).

Each data type (for example, image, text, voice, etc.) has a number of methods defined for actions such as capture or display. These methods are stored in the file entry and are invoked by a message process. This mechanism allows us to support different workstation hardware without change to the application software.

Factors Influencing Integration Scheme

The selection of a method of interfacing with outside systems or software depends on a number of factors including:

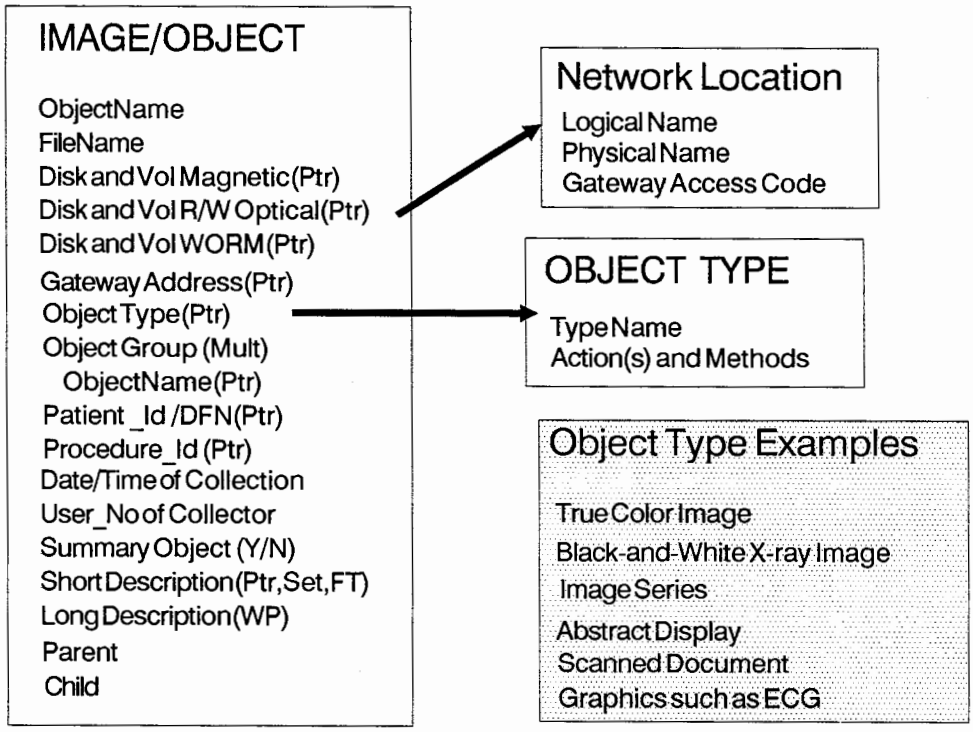

Figure 2: An object-oriented file structure is used to support data access, interrelationships of data, and integration of processes. The object file and network location files are shown here.

- Whether the interface software is being developed by simultaneously for both systems, or whether the other systems is previously existing with a predefined interface

- Availability of the various components in the interface environment, such as ZCALL function capability, operating system support for communications mechanisms, and network servers;

- Interface performance requirements;

- Flexibility desired of the interface.

When interfaces are being created to allow two existing systems to communicate, the technique of writing messages to a shared disk drive or server can ease the development process as it allows separation of development and testing of the two sides of the interface. It also creates a transaction record that is useful in debugging. This approach is also beneficial when messages are large.

System interfaces require agreement on the definition of messages. In some cases these are based on a standard, such as ACR-NEMA messages; in others they must be mutually defined. Major advantages of using an interface supported by an operating system include the "standard" message format and interface capabilities that it provides for industry developers. These features enable large numbers of outside software packages to become available to MUMPS users.

Conclusions

A number of mechanisms have been used to allow MUMPS to interoperate with non-MUMPS systems and processes. The ability to control non-MUMPS software provides new capabilities to MUMPS. This ability is critical to allow MUMPS applications to compete with other commercial systems in the future.

References

1. Novell Netware Version 4.0.

2. Dayhoff Ruth E, Kuzmak Peter, Maloney Daniel. Medical images as an integral part of the patient's automated record, Proc. of Symposium on Computer Assisted Radiology, June 1992.

3. Kuzmak Peter, Dayhoff Ruth (final paper title for 1993 MUG), Proceedings of the 1993 Annual Meeting of the M Technology Association.

4. DataTree Windows Version 1.0 Manual, p 45, June 1992.

5. Dayhoff Ruth, Maloney Daniel. Exchange of Veterans Affairs Medical Data Using National and Local Networks, Annals of the New York Academy of Sciences "Extended Clinical Consulting by Hospital Computer Networks", Vol.670, 1993, pp. 62-63.

6. Maloney Daniel, Dayhoff Ruth. Multimedia Object File Design for Medical Images, Proc. of the 1991 MUMPS Users Group Annual Meeting, New Orleans, June 1991.

7. Dayhoff Ruth E, Maloney Daniel L, Majurski William J. Porting the DHCP Imaging System to an X Window Environment Using Object-Oriented File Structures, Proc. MUMPS Users' Group Annual Meeting, 1992.

8. Decentralized Hospital Computer Program, Dept. of Veterans Affairs, 1993.