Introduction

This manual provides information about the structure of the Veterans Health Information Systems and Technology Architecture (**V**_IST_**A**) software known as the Remote Procedure Call (RPC) Broker (also referred to as "Broker"). This manual consists of technical material specifically intended for **V**_IST_**A** systems managers and developers.

PRODUCT OVERVIEW

The RPC Broker is considered to be part of the infrastructure of **V**_IST_**A**. It establishes a common and consistent foundation for communication between clients and **V**_IST_**A** M servers.

The RPC Broker is a bridge connecting the client application front-end on the workstation (e.g., Delphi GUI applications) to the M-based data and business rules on the server. It links one part of a program running on a workstation to its counterpart on the server. The client and the server can be, and most often are, written in different computer languages. Therefore, the RPC Broker bridges the gap between the traditionally proprietary **V**_IST_**A** and COTS/HOST products.

The RPC Broker includes:

- A common communications driver for the M server interface that handles the device-specific characteristics of the supported communications protocol.

- An interface component on the M server, separate from the communications driver, that interprets client messages, executes the required code, and eventually returns data to the communications driver.

- A common file on the M server which all applications use to store the information about the queries to which they respond (i.e., REMOTE PROCEDURE file [#8994]).

- The Client Agent application that runs on client workstations, supporting single signon.

- The TRPCBroker component for Delphi, enabling development of client applications that can communicate via the RPC Broker.

- A dynamic link library (DLL) that provides access to RPC Broker functionality for development environments other than Delphi.

RELATED MANUALS AND OTHER REFERENCES

Readers who wish to learn more about the RPC Broker should consult the following:

- *RPC Broker V. 1.1 Getting Started with the Broker Development Kit*

- *RPC Broker V. 1.1 Installation Guide*

- *RPC Broker V. 1.1 Release Notes*

- *RPC Broker V. 1.1 Systems Manual*

- *Online RPC Broker Developer's Guide* (i.e., BROKER.HLP)

- The *MIRMO/ISC Operations Document*, "Chapter 10"

- *Programming Standards and Conventions (SAC)*

- RPC Broker Home Page at the following web address:

    http://www.vista.med.va.gov/softserv/infrastr.uct/broker/index.html

    This site contains additional information and documentation (e.g., Frequently Asked Questions [FAQs]) available in Hypertext Markup Language (HTML).

Broker documentation is made available online, on paper, and in Adobe Acrobat Portable Document Format (.PDF). The .PDF documents must be read using the Adobe Acrobat Reader (i.e., ACROREAD.EXE), which is freely distributed by Adobe Systems Incorporated at the following web address:

http://www.adobe.com/

For more information on the use of the Adobe Acrobat Reader, please refer to the "Adobe Acrobat Quick Guide" at the following web address:

http://www.vista.med.va.gov/softserv/infrastr.uct/acrobat/index.html

Implementation and Maintenance

The *RPC Broker V. 1.1 Installation Guide* provides detailed information regarding the installation of the RPC Broker. It also contains many requirements and recommendation regarding how the Broker should be configured. Be sure to read the Installation Guide before attempting to install the RPC Broker.

SITE PARAMETERS

This topic lists the site parameters that can be set to customize the operation of the RPC Broker.

The following two areas of the Broker require site parameter review and configuration:

- Broker Listeners

- Single Signon Functionality

Broker Listeners

The RPC BROKER SITE PARAMETERS file (#8994.1) includes a LISTENER field and various subfields. The LISTENER field is a multiple and should contain all Listeners that you plan to run. A simple change of the STATUS subfield from STOPPED to START will start that particular Listener. Conversely, you can STOP a RUNNING Listener the same way.

For more information about configuring Broker Listeners, please refer to the RPC Broker V. 1.1 Systems Manual.

Single Signon

Single signon means that when a user has entered their access and verify code once to create an active connection to the server, they don't need to enter them again to make additional connections. Connections include GUI applications, like CPRS, and also the traditional roll-and-scroll system (through telnet).

Fields in the NEW PERSON file (#200) and the KERNEL SYSTEM PARAMETERS file (#8989.3) control access to the single signon feature.

For more information about configuring site parameters for single signon, please refer to the RPC Broker V. 1.1 Systems Manual.

PERFORMANCE AND SCALABILITY

Current performance statistics are limited. However, results indicate that the processing time and resources consumed by the Broker itself are minimal. The RPC Broker doesn't introduce any additional overhead to the messages sent between the client and the server.

Performance should be measured at the application level to determine the amount of resources consumed by **V**_IST_**A** client/server applications that use the Broker.

We anticipate collecting more data from this release of the Broker to provide us with comparison statistics on performance and scalability in a production environment that we can pass on to all users of the Broker.

File List

M SERVER FILES

The RPC Broker consists of a single global with two files. This chapter describes the RPC Broker files including the file number, file name, global location, and description of the files.

File #	File Name	Global Location
8994	REMOTE PROCEDURE	^XWB(8994,
8994.1	RPC BROKER SITE PARAMETERS	^XWB(8994.1,

Table 2: RPC Broker Files

REMOTE PROCEDURE File (#8994)

This file is used as a repository of server-based procedures (i.e., remote procedure calls [RPCs]) in the context of the Client/Server architecture. All RPCs used by any site-specific client/server application software using the RPC Broker interface must be registered and stored in this file. Applications running on client workstations can invoke (call) the RPCs in this file to be executed by the server and the results are returned to the client application. Each RPC is associated with an entry point (i.e., ROUTINE with optional TAG).

The RPC subfield (#19.05) of the OPTION File (#19) points to RPC field (#.01) of the REMOTE PROCEDURE file (#8994).

Data is not distributed with this file. RPCs are distributed and installed as separate components during the installation of the RPC Broker, however.

RPC BROKER SITE PARAMETERS file (#8994.1)

Site managers can use this file to configure and adjust many characteristics of an RPC Broker installation at a site.

Data is not distributed with this file.

If you are an MSM 4.3.0 site or greater and using MSERVER instead of the Broker Listener, the current functionality provided by this file is not applicable to your site.

CLIENT FILES

End-User Workstation

\Program Files\Vista\Broker

```
Clagent.exe
Clagent.hlp
RPCTest.exe
RPCTest.hlp
```

\Windows\System(32)

```
Bapi32.dll
VistaBroker.dpl
```

Programmer Workstation

\Program Files\Vista\Broker

```
BrokerProgPref.exe
BrokerProgPref.hlp
ServerList.exe
Serverlsit.hlp
```

\Program Files\Vista\Bdk32\D2

```
FrmSignonMessage.dcu
FrmSignonMessage.dfm
Hash.dcu
Loginfrm.dcu
Loginfrm.dfm
MfunStr.dcu
RpcbEdtr.dcu
RpcbErr.dcu
Rpcberr.dfm
RpcConf1.dcu
Rpcconf1.dfm
rpcnet.dcu
Rpcnet.dfm
SgnonCnf.dcu
SgnonCnf.dfm
SplVista.dcu
Splvista.dfm
Trpcb.dcr
Trpcb.dcu
VCEdit.dcu
Vcedit.dfm
WSockc.dcu
XWBut1.dcu
```

\Program Files\Vista\Bdk32\D3

```
FrmSignonMessage.dcu
FrmSignonMessage.dfm
Hash.dcu
Loginfrm.dcu
Loginfrm.dfm
MfunStr.dcu
RpcbEdtr.dcu
RpcbErr.dcu
Rpcberr.dfm
RpcConf1.dcu
Rpcconf1.dfm
rpcnet.dcu
Rpcnet.dfm
SgnonCnf.dcu
SgnonCnf.dfm
SplVista.dcu
Splvista.dfm
Trpcb.dcu
VCEdit.dcu
Vcedit.dfm
VistABroker.dcp
VistABroker.dpl (same file as in the \Windows\System directory)
WSockc.dcu
XWBut1.dcu
```

\Program Files\Vista\Bdk32\Doc

```
Broker.cnt
Broker.hlp
Broker.kwf
BrokerSM.cnt
BrokerSM.hlp
xwb1_1DG.PDF
xwb1_1IG.PDF
xwb1_1RN.PDF
xwb1_1SM.PDF
xwb1_1TM.PDF
```

\Program Files\Vista\Bdk32\Headers

```
BAPI32.bas
Bapi32.h
Bapi32.hpp
```

\Program Files\Vista\Bdk32\Samples\BrokerEx

```
BrokerExample.DPR
BrokerExampleAboutFrm.DFM
BrokerExampleAboutFrm.PAS
BrokerExampleFrm.DFM
BrokerExampleFrm.PAS
```

\Program Files\Vista\Bdk32\Samples\Vb5egcho

```
BAPI32.bas
egcho.bas
Egcho.frm
egcho.vbp
```

File List

Global Translation, Journaling, and Protection

TRANSLATION

Translation is recommended for the sole RPC Broker global (i.e., ^XWB global). The ^XWB global has the potential to be read-intensive as more and more remote procedures are added to it in the future.

For DSM and OpenM Systems:

It is best to translate the global to a volume set other than ROU. In order for translation to take effect on DSM and OpenM systems, DSM and OpenM must be rebooted.

Cookbook recommendations should also be consulted for suggestions regarding journaling, translation, and replication; the information here may not apply:

- *DSM for OpenVMS sites should consult the most recent Computer Operations Management and Procedures for AXP Systems (COMPAS) manual.*

- *OpenM sites should refer to the "Configuring OpenM" chapter in the VISTA-NT Conversion Checklist.*

For MSM Systems:

It is best to translate the global to a file server.

Cookbook recommendations should also be consulted. MSM sites should consult the most recent 486 Cookbook and MSM System Managers Guide for instructions and recommendations regarding journaling, translation, and replication; the information here may not apply.

JOURNALING

Journaling of this global is not required, since the ^XWB global, for the most part is static (except during the addition of new remote procedures).

PROTECTION

The following global protection should be set:

Global Name	Protection		
	DSM for OpenVMS	OpenM	MSM-DOS
^XWB	System: RWD	Owner: RWD	System: RWD
	World: RW	Group: N	World: RWD
	Group: RW	World: N	Group: RWD
	User: RW	Network: RWD	User: RWD

Table 3: RPC Broker Global Information

Routine List

This chapter contains a list of the routines exported with the RPC Broker. A brief description of the routines is provided.

Routine	Description
XWBBRK	This routine contains calls that are designed to parse the various attributes of the Broker messages. All of this information is used internally. In the case of large arrays sent by the client, the function BREAD is used to read in the variable length subscripts and values.
XWBBRK2	This routine is a continuation of XWBBRK. The main entry point (i.e., CAPI) actually calls the application RPC.
XWBCAGNT	Server code for RPC Broker client agent application.
XWBEXMPL	This routine is used to support the Broker Example application. The Broker Example application is used to test the RPC Broker connectivity, actions, and RPCs. It is distributed with the Broker.
XWBFM	This routine contains entry points used to interface to the VA FileMan database server.
XWBLIB	This routine contains various functions and procedures used by the Broker. It is best described as a library or depository.
XWBSEC	This routine contains various functions and procedures used by the Broker. Calls in this routine are used for client/server security.
XWBTCP	This routine contains functions and procedures used to control the Broker TCP/IP Listener process. Systems personnel can use calls in this routine to start, stop, and debug the Broker process.
XWBTCPC	This job is started for each Broker request. The Listener process (i.e., XWBTCPL) will receive a connection request from a client and then dispatch, using the M JOB command, XWBTCPC to manage the rest of the interaction.
XWBTCPL	This is the Broker Listener process. IRM starts this job. It remains running on a system listening for TCP/IP connection requests. Once a request is received, this routine will start a separate process to manage the rest of the connection, then returns to "listening" for a new request.

Routine	Description
XWBZ1	This routine is used to support the Echo application. The Echo application is used to test the RPC Broker connectivity, actions, and APIs. It is distributed with the Broker.

ROUTINE MAPPING

RPC Broker routines are *not* required to be mapped to any account.

Exported Options

The following options are exported with the RPC Broker:

Name	Menu Text	Type
XWB BROKER EXAMPLE	RPC BROKER PROGRAMMING EXAMPLE	Broker (Client/Server)
XWB EGCHO	RPC BROKER DEMO/TEST	Broker (Client/Server)
XWB LISTENER STARTER	Start RPC Broker Listeners	Run Routine
XWB RPC TEST	RPC	Broker (Client/Server)

Table 4: Exported Options

Client/server applications are a new type of option (i.e., Type "B", Broker client/server options) in the OPTION file (#19). The user must have the client/server application option assigned to them as with any other assigned option in **V**_IST_**A**. The client/server application will only run for those users who are allowed to activate it.

The client/server application options will not be displayed in the user's menu tree.

XWB BROKER EXAMPLE

This option supports the Broker Example demonstration program provided in the Broker Development Kit. Developers should assign this option to themselves, if they want to try out the Broker Example application. For programmers who have the XUPROGMODE key, however, assigning this option to themselves is not necessary.

XWB EGCHO

This option supports the EGCHO demonstration program provided in the Broker Development Kit. Developers should assign this option to themselves, if they want to try out the EGCHO application. For programmers who have the XUPROGMODE key, however, assigning this option to themselves is not necessary.

XWB LISTENER STARTER

The XWB LISTENER STARTER option can be used to start one or more Broker Listeners at one time. It works in combination with the new CONTROLLED BY LISTENER STARTER (#2) subfield of the Port (#1) subfield of the Listener (#7) multiple field in the RPC BROKER SITE PARAMETERS file (#8994.1). The CONTROLLED BY LISTENER STARTER field is a Yes/No set of codes type field. All of the Listener entries in this file that have CONTROLLED BY LISTENER STARTER set to Yes will be started when the XWB LISTENER STARTER option is run.

Additionally, the XWB LISTENER STARTER option can be scheduled through TaskMan such that whenever TaskMan starts up, the listener processes are started. To do this, schedule the XWB LISTENER STARTER option with SPECIAL QUEUING set to STARTUP.

This option should only be made available to system managers.

XWB RPC TEST

It is recommended that the XWB RPC TEST option be given to users running Broker-based **V**_IST_**A** client/server applications. The RPCTEST.EXE program on the client workstation runs the RPC Broker Diagnostic Program. This tool can be used to verify and test the Broker client/server connection and signon process. It displays information about the client and the server and can be a useful debugging tool for IRM.

To enable remote troubleshooting by IRM for all users, you can put this option on the Common menu (i.e., System Command Options menu [XUCOMMAND]). This enables any user to run the RPCTEST.EXE program on their workstation at your request.

EXPORTED RPCS

The RPC Broker distributes the following remote procedure calls (RPCs):

    XWB CREATE CONTEXT
    XWB EGCHO BIG LIST
    XWB EGCHO LIST
    XWB EGCHO MEMO
    XWB EGCHO SORT LIST
    XWB EGCHO STRING
    XWB EXAMPLE ECHO STRING
    XWB EXAMPLE GET LIST
    XWB EXAMPLE SORT NUMBERS
    XWB EXAMPLE WPTEXT

XWB GET VARIABLE VALUE
XWB FILE LIST
XWB FILENAME CHECK
XWB GET VARIABLE VALUE
XWB RPC LIST

Exported Options

Archiving and Purging

ARCHIVING

There are no package-specific archiving procedures or recommendations for the RPC Broker ^XWB global or the REMOTE PROCEDURE and RPC BROKER SITE PARAMETERS files.

PURGING

There are no package-specific purging procedures or recommendations for the RPC Broker ^XWB global or the REMOTE PROCEDURE and RPC BROKER SITE PARAMETERS files.

Archiving and Purging

Callable Routines

The RPC Broker does *not* provide any callable M routines. However, other programming interfaces are provided (e.g., Delphi component, DLL, Pascal functions, and RPCs).

For information on these other programming interfaces, please refer to the "External Interfaces" chapter in this manual.

RPC Broker V. 1.1 Technical Manual

External Interfaces

The following external interfaces to RPC Broker functionality are provided:

TRPCBROKER COMPONENT

The TRPCBroker component provides all functionality needed for client applications to communicate with **V**IST**A** M servers via the RPC Broker. The TRPCBroker component is compatible with Borland Delphi 2.0 and greater.

For more information on the TRPCBroker component, please refer to the RPC Broker V. 1.1 Getting Started with the Broker Development Kit and Online RPC Broker Developer's Guide manuals.

TRPCBROKER DYNAMIC LINK LIBRARY (DLL)

The TRPCBroker DLL (BAPI32.DLL) provides access to RPC Broker functionality for development environments other than Delphi.

For more information on the TRPCBroker DLL, please refer to the RPC Broker V. 1.1 Getting Started with the Broker Development Kit and Online RPC Broker Developer's Guide manuals.

PASCAL FUNCTIONS

The following Pascal functions are provided by the TRPCBroker component:

- GetServerInfo function
- Splash Screen functions: SplashOpen and SplashClose
- Piece function
- Translate function
- Encryption functions: Decrypt and Encrypt

For more information on these Pascal functions, please refer to the RPC Broker V. 1.1 Getting Started with the Broker Development Kit and Online RPC Broker Developer's Guide manuals.

RPC BROKER REMOTE PROCEDURES

The following RPC is provided for use by developers:

- XWB GET VARIABLE VALUE

For more information, please refer to the RPC Broker V. 1.1 Getting Started with the Broker Development Kit and Online RPC Broker Developer's Guide manuals.

External Relations

RELATIONSHIP TO OTHER PACKAGES

The RPC Broker software has been developed to aid the **V**_IST_**A** development community and Information Resources Management (IRM) and is considered to be part of the infrastructure of **V**_IST_**A**. Other infrastructure products include VA FileMan, Kernel, and MailMan. The RPC Broker will be used by all client applications written as part of **V**_IST_**A**. The RPC Broker fully integrates with VA FileMan V. 21.0 and Kernel V. 8.0.

The absence of RPC Broker software on an M server will disable the functioning of any client application that depends on the RPC Broker to communicate with the M Server.

It is possible that the use of RPCs will also be extended to non-client applications. In this case, the REMOTE PROCEDURE FILE must be present for those applications to function correctly.

RELATIONSHIP WITH KERNEL AND VA FILEMAN

Before installing the RPC Broker, Kernel V. 8.0, Kernel Toolkit V. 7.3, and VA FileMan V. 21.0 must be in place and fully patched.

RELATIONSHIPS WITH OPERATING SYSTEMS

On the client side, it was decided that the 32-bit Microsoft Windows environment would be the supported platform. Thus, the client portions of the RPC Broker are compatible with Microsoft Windows 95 or higher, and Microsoft Windows NT 3.5 or higher.

On the server side, the RPC Broker supports the following ANSI M environments:

- Digital Standard M (DSM) V6.3-031 for OpenVMS AXP or greater

- InterSystems OpenM for NT version 7

- Micronetics Standard M (MSM) for Windows NT, V. 4.2.4 or greater

DBA APPROVALS and DATABASE INTEGRATION AGREEMENTS (DBIAs)

To obtain the current list of DBIAs that the RPC Broker is a custodian of:

1. Sign on to the Forum system (forum.va.gov).
2. Go to the DBA menu.
3. Select the Integration Agreements menu.
4. Select the Custodial Package menu.
5. Choose the ACTIVE by Custodial Package option.
6. When this option prompts you for a package, enter RPC BROKER.
7. All current DBIAs for which the RPC Broker package is custodian are listed.

To obtain detailed information on a specific integration agreement:

1. Sign on to the Forum system (forum.va.gov).
2. Go to the DBA menu.
3. Select the Integration Agreements menu.
4. Select the Inquire option.
5. When prompted for "INTEGRATION REFERENCES", enter the integration agreement number of the DBIA you would like to display.
6. The option then lists the full text of the DBIA you requested.

To obtain the current list of DBIAs that the RPC Broker is a subscriber to:

1. Sign on to the Forum system (forum.va.gov).

2. Go to the DBA menu.

3. Select the Integration Agreements menu.

4. Select the Subscriber Package menu.

5. Choose the Print ACTIVE by Subscribing Package option.

6. When prompted "START WITH SUBSCRIBING PACKAGE", enter RPC BROKER (in uppercase). When prompted "GO TO SUBSCRIBING PACKAGE", enter RPC BROKER (in uppercase).

7. All current DBIAs to which the RPC Broker package is a subscriber are listed.

External Relations

Internal Relations

No options in the RPC Broker product assume that the entry/exit logic of another option has already occurred.

Package-wide Variables

The RPC Broker does *not* create any package-wide variables that have received *Programming Standards and Conventions Committee (SACC)* exemptions.

Package-wide Variables

Software Product Security

SECURITY MANAGEMENT

There are no special legal requirements involved in the use of the RPC Broker product.

MAIL GROUPS AND ALERTS

The RPC Broker does not make use of mail groups or alerts.

REMOTE SYSTEMS

The M server process of the RPC Broker allows connections from client applications. Connection by those client applications is subject to authentication as any normal logon requires. Client applications can use any remote procedure call (RPC) authorized to the application, if the application is authorized to the signed-on user. Data is typically exchanged between clients and the RPC Broker server. Clients can be anywhere on VA's TCP/IP network.

Encryption is used when a user's access and verify codes are sent from the client to the server.

In addition, an encryption API is provided for developer use in their own applications to encode and decode messages passed between client and server.

Security with the RPC Broker is a four-part process:

1. Client workstations must send a valid connection request to the M Server.

2. Users must have valid Access and Verify codes.

3. Users must be valid users of a **V**ISTA client/server application.

4. Any remote procedure call must be registered and valid for the application being executed.

For more information regarding Broker security, please refer to Chapter 2, "Security", in the RPC Broker V. 1.1 Systems Manual.

INTERFACING

No non-VA products are embedded in or required by the RPC Broker, other than those provided by the underlying operating systems.

ELECTRONIC SIGNATURES

Electronic signatures are not used within the RPC Broker.

SECURITY KEYS

There are *no* specific security keys exported with the RPC Broker software. However, to bypass security for development purposes, we recommend client/server application developers be assigned the XUPROGMODE security key.

All users assigned the XUPROGMODE security key can do the following:

- Run any **V**_IST_**A** client/server application regardless of whether it is in their menu tree or not, and

- Access any RPC without regard to the application context.

FILE SECURITY

The RPC Broker establishes the following security over its files:

Number	Name	DD	RD	WR	DEL	LAYGO	AUDIT
8994	REMOTE PROCEDURE	@	@	@	@	@	@
8994.1	RPC BROKER SITE PARAMETERS	@	@	@	@	@	@

Table 5: File Security

OFFICIAL POLICIES

Modification of any part of the RPC Broker software is strongly discouraged.

Distribution of the RPC Broker software is unrestricted.

Glossary

ACCESS CODE | A code that, along with the verify code, allows the computer to identify you as a user authorized to gain access to the computer. Your code is greater than six and less than twenty characters long; can be numeric, alphabetic, or a combination of both; and is usually assigned by a site manager or application coordinator. It is used by the Kernel's Sign-on/Security system to identify the user (see Verify Code).

ALERTS | Brief online notices that are issued to users as they complete a cycle through the menu system. Alerts are designed to provide interactive notification of pending computing activities, such as the need to reorder supplies or review a patient's clinical test results. Along with the alert message is an indication that the View Alerts common option should be chosen to take further action.

ANSI MUMPS | The MUMPS programming language is a standard, that is an **A**merican **N**ational **S**tandard (ANS). MUMPS stands for **M**assachusetts **U**tility **M**ulti-**p**rogramming **S**ystem and is abbreviated as M.

APPLICATION PROGRAMMING INTERFACE (API) | Programmer calls provided by the Kernel for use by application programmers. APIs allow programmers to carry out standard computing activities without needing to duplicate Kernel utilities in their own packages. APIs also further DBA goals of system integration by channeling activities, such as adding new users, through a limited number of callable entry points.

ARRAY | An arrangement of elements in one or more dimensions. An M array is a set of nodes referenced by subscripts that share the same variable name.

CALLABLE ENTRY POINT | An authorized programmer call that may be used in any **V**_IST_**A** application package. The DBA maintains the list of DBIC-approved entry points.

CLIENT | A single term used interchangeably to refer to the user, the workstation, and the portion of the program that runs on the workstation. In an object-oriented environment, a client is a member of a group that uses the services of an unrelated group. If the client is on a local area network (LAN), it can share resources with another computer (server).

COMPONENT	An object-oriented term used to describe the building blocks of GUI applications. A software object that contains data and code. A component may or may not be visible. These components interact with other components on a form to create the GUI user application interface.
COTS	**C**ommercial **O**ff-**t**he-**S**helf. COTS refers to software packages that can be purchased by the public and used in support of **V***IST***A**.
DATA DICTIONARY	The Data Dictionary is a global containing a description of what kind of data is stored in the global corresponding to a particular file. The data is used internally by VA FileMan for interpreting and processing files.
	A Data Dictionary (DD) contains the definitions of a file's elements (fields or data attributes), relationships to other files, and structure or design. Users generally review the definitions of a file's elements or data attributes; programmers review the definitions of a file's internal structure.
DBIA	**D**atab**a**se **I**ntegration **A**greement, a formal understanding between two or more application packages which describes how data is shared or how packages interact. The DBA maintains a list of DBIAs between package developers, allowing the use of internal entry points or other package-specific features that are not available to the general programming public.
DIRECT MODE UTILITY	A programmer call that is made when working in direct programmer mode. A direct mode utility is entered at the M prompt (e.g., >**D ^XUP**). Calls that are documented as direct mode utilities *cannot* be used in application package code.

DLL

Dynamic **L**ink **L**ibrary. A DLL allows executable routines to be stored separately as files with a DLL extension. These routines are only loaded when a program calls for them. DLLs provide several advantages:

1. DLLs help save on computer memory, since memory is only consumed when a DLL is loaded.

2. DLLs ease maintenance tasks. Because the DLL is a separate file, any modifications made to the DLL will not affect the operation of the calling program or any other DLL.

3. DLLs help avoid redundant routines. They provide generic functions that can be utilized by a variety of programs.

ERROR TRAP

A mechanism to capture system errors and record facts about the computing context such as the local symbol table, last global reference, and routine in use. Operating systems provide tools such as the %ER utility. The Kernel provides a generic error trapping mechanism with use of the ^%ZTER global and ^XTER* routines. Errors can be trapped and, when possible, the user is returned to the menu system.

FORUM

The central e-mail system within **V***IST***A**. Developers use FORUM to communicate at a national level about programming and other issues. FORUM is located at the Washington, DC CIO Field Office (162-2).

GUI

Graphical **U**ser Interface. A type of display format that enables users to choose commands, initiate programs, and other options by selecting pictorial representations (icons) via a mouse or a keyboard.

IRM

Information **R**esource **M**anagement. A service at VA medical centers responsible for computer management and system security.

KERNEL

A set of M software routines that function as an intermediary between the host operating system and the **V***IST***A** application packages enabling packages to coexist in a standard OS-independent computing environment. The Kernel provides a standard and consistent user and programmer interface between application packages and the underlying M implementations.

MAILMAN	The Kernel module that provides a mechanism for handling electronic communication, whether it is user-oriented mail messages, automatic firing of bulletins, or initiation of server-handled data transmissions.
MENU MANAGER	The Kernel module that controls the presentation of user activities such as menu choices or options. Information about each user's menu choices is stored in the Compiled Menu System, the ^XUTL global, for easy and efficient access.
METHOD	An object-oriented term used to describe procedures and functions, also referred to as routines, associated with a particular component. Methods are called at run time. They are never activated at design time.
MULTIPLE	A multiple-valued field; a subfile. In many respects, a multiple is structured like a file.
MUMPS (ANSI STANDARD)	A programming language recognized by the **A**merican **N**ational **S**tandards **I**nstitute (ANSI). The acronym MUMPS stands for **M**assachusetts General Hospital **U**tility **M**ulti-**p**rogramming **S**ystem and is abbreviated as M.
NAMESPACING	A convention for naming **V**_IST_**A** package elements. The Database Administrator (DBA) assigns unique character strings for package developers to use in naming routines, options, and other package elements so that packages may coexist. The DBA also assigns a separate range of file numbers to each package.
NODE	In a tree structure, a point at which subordinate items of data originate. An M array element is characterized by a name and a unique subscript. Thus the terms: node, array element, and subscripted variable are synonymous. In a global array, each node might have specific fields or "pieces" reserved for data attributes such as name.
OPTION	As an item on a menu, an option provides an opportunity for users to select it, thereby invoking the associated computing activity. In **V**_IST_**A**, an entry in the OPTION file (#19). Options may also be scheduled to run in the background, non-interactively, by TaskMan.

PROPERTIES
An object-oriented term used to describe the attributes associated with components on a GUI form. These attributes, collectively, indicate how a component is to be displayed to the user in the application interface. Properties are activated at design time.

REMOTE PROCEDURE CALL
A remote procedure call (RPC) is essentially M code that may take optional parameters to do some work and then return either a single value or an array back to the client application.

ROUTINE
A program or a sequence of instructions called by a program that may have some general or frequent use. M routines are groups of program lines that are saved, loaded, and called as a single unit via a specific name.

SAC
Standards **a**nd **C**onventions. Through a process of verification, **V**_IST_**A** packages are reviewed with respect to SAC guidelines as set forth by the Standards and Conventions Committee (SACC). Package documentation is similarly reviewed in terms of standards set by the Documentation Standards and Conventions Committee (DSCC).

SACC
V_IST_**A S**tandards **a**nd **C**onventions **C**ommittee. This Committee is responsible for maintaining the document called SAC.

SECURITY KEY
The purpose of Security Keys is to set a layer of protection on the range of computing capabilities available with a particular software package. The availability of options is based on the level of system access granted to each user.

SERVER
The computer where the data and the Business Rules reside. It makes resources available to client workstations on the network. In **V**_IST_**A**, it is an entry in the OPTION file (#19). An automated mail protocol that is activated by sending a message to a server at another location with the "S.server" syntax. A server's activity is specified in the OPTION file (#19) and can be the running of a routine or the placement of data into a file.

SIGN-ON/SECURITY
The Kernel module that regulates access to the menu system. It performs a number of checks to determine whether access can be permitted at a particular time. A log of signons is maintained.

SUBSCRIPT	A symbol that is associated with the name of a set to identify a particular subset or element. In M, a numeric or string value that: is enclosed in parentheses, is appended to the name of a local or global variable, and identifies a specific node within an array.
UCI	**U**ser **C**lass **I**dentification, a computing area. The MGR UCI is typically the Manager's account, while VAH or ROU may be Production accounts.
VA FILEMAN (FILE MANAGER)	A set of programs used to enter, maintain, access, and manipulate a database management system consisting of files. A package of online computer routines written in the M language which can be used as a stand-alone database system or as a set of application utilities. In either form, such routines can be used to define, enter, edit, and retrieve information from a set of computer stored files.
VERIFY CODE	The Kernel's Sign-on/Security system uses the verify code to validate the user's identity. This is an additional security precaution used in conjunction with the Access Code. Like the Access Code, it is also 6 to 20 characters in length. If entered incorrectly, it does not allow the user to access the computer. To protect the user, both codes are invisible on the terminal screen.
VISTA	**V**eterans Health **I**nformation **S**ystems and **T**echnology **A**rchitecture. **V**_IST_**A** includes the VA's application software (i.e., Windows-based and locally-developed applications, roll-and-scroll, and interfaces such as software links to commercial packages). In addition, it encompasses the VA's uses of new automated technology including the clinical workstations. **V**_IST_**A** encompasses the rich automated environment already present at local VA medical facilities.

Index

RPC BROKER

TECHNICAL MANUAL

Version 1.1

September 1997

Table of Contents

Orientation

HOW TO USE THIS MANUAL

This manual uses several methods to highlight different aspects of the material:

■ All uppercase is reserved for the representation of M code, variable names, or the formal name of options, field and file names, and security keys (e.g., the XUPROGMODE key).

■ Various symbols are used throughout the documentation to alert the reader to special information. The following table gives a description of each of these symbols:

Symbol	Description
	Used to inform the reader of general information including references to additional reading material
	Used to caution the reader to take special notice of critical information

Table 1: Documentation Symbol Descriptions

ASSUMPTIONS ABOUT THE READER

This manual is written with the assumption that the reader is familiar with the following:

• **V**_IST_**A** computing environment (e.g., Kernel Installation and Distribution System [KIDS])

• VA FileMan data structures and terminology

• Microsoft Windows

• M programming language

HOW TO OBTAIN TECHNICAL INFORMATION ONLINE

Technical information about the RPC Broker is available online. It can be obtained in a number of ways as described below.

List File Attributes

The VA FileMan List File Attributes option allows you to generate documentation pertaining to files and file structure stored in the Data Dictionary.

Listing the data dictionary in Standard format yields the following Data Dictionary information for a specified file (e.g., REMOTE PROCEDURE file [#8994]):

- File name and description
- Fields in the file, including descriptions
- Identifiers
- Cross-references
- Files pointed to by the file specified
- Files that point to the file specified
- Input templates
- Print templates
- Sort templates

Listing the data dictionary in Global Map format generates a list of all cross-references for the file selected, global location of each field in the file, input templates, print templates, and sort templates.

For more information regarding VA FileMan Data Dictionary options, please refer to Chapter 1, "How to Display Information About Files", in the VA FileMan V. 21.0 User Manual.

KIDS Build File

You can list the server-side components that are distributed with the RPC Broker by using the Build File Print option. The menu path to that option is:

```
Programmer Options ... [XUPROG]
  KIDS   Kernel Installation & Distribution System ... [XPD MAIN]
    Utilities ... [XPD UTILITY]
      Build File Print [XPD PRINT BUILD]
```