RPC Broker 1.1

Release Notes

February 2017

Department of Veterans Affairs (VA)

Office of Information and Technology (OI&T)

Enterprise Program Management Office (EPMO)

Revision History

Documentation Revisions

Date	Revision	Description	Authors
02/16/2017	7.0	Tech Edits based on release of RPC Broker Patch XWB*1.1*65: - Reformatted document to follow current documentation standards and style formatting requirements. - Reformatted all sections to follow the most current Release Notes template Version 1.1, dated July 2016. - Added/Updated support for 2-factor authentication, Microsoft® Windows 32-bit Client applications, Delphi supported versions, added TXWBSSOiToken, and updated patch references in Section 4.1. - Added Section 4.1.1, "2-Factor Authentication." - Updated Sections 4.1.3; changed references from "Attachmate Reflections" to "Micro Focus Reflection." - Updated Section 4.1.12; added TXWBSSOiToken and alphabetized the list of components. - Added Section 4.1.12.5, "TXWBSSOiToken." - Updated Section 4.1.13; added TXWBSSOiToken. - Updated Section 4.1.15.2. - Added Section 4.1.15.6. - Updated Section 4.2.2; removed Note. - Added Section 4.3. **RPC Broker 1.1; XWB*1.1*65 BDK**	- Developer H. W. - Technical Writer: T. B.
04/27/2016	6.0	Tech Edits based on release of RPC Broker Patch XWB*1.1*60 (released 06/11/2015): - Reformatted document to follow current documentation standards and style formatting requirements. - Updated Section 1.1. - Added Section 2.1.	- Developer H. W. - Technical Writer: T. B.

Date	Revision	Description	Authors
		• Added Section 3.1.1. • Updated Section 3.1.2. • Deleted Section 3.1.4, "Full Backward Compatibility with Broker 1.0", since there is no means of testing this. The only Broker 1.0 application was PCMM, and the most recently released PCMM version no longer uses Broker 1.0. • Updated Section 3.2.1. • Updated Section 4. • Updated Section 4.2; deleted references to the TSharedBroker and TSharedRPCBroker components. • Deleted Section 4.2.3, "TSharedBroker" and Section 4.2.4, "TSharedRPCBroker." • Updated Section 4.2.6.2. • Updated Section 4.2.7.2. • Updated Section 4.2.7.3. • Updated Figure 1. • Added Note to Section 4.5. • Updated Sections 5.1 and 5.2 for Broker Help file references. • Updated Figure 2. • Updated references to show RPC Broker Patch XWB*1.1*60 supports Delphi XE7, XE6, XE5, and XE4 throughout. • Updated help file references from "BROKER.HLP" to "Broker_1_1.chm" throughout. **RPC Broker 1.1**	
12/04/2013	5.1	Tech Edit: • Updated document for RPC Broker Patch XWB*1.1*50 based on feedback from H Westra. • Removed references related to Virgin Installations throughout. • Updated file name references throughout. • Removed distribution files that are obsolete or no longer distributed throughout. • Updated RPC Broker support on the following software:	• Developer: H. W. • Technical Writer: T. B.

Date	Revision	Description	Authors
		o Microsoft® XP and 7 (operating system) throughout. o Microsoft® Office Products 2010 throughout. o Changed references from “Borland” to “Embarcadero” and updated support for Delphi Versions XE5, XE4, XE3, and XE2 throughout. • Updated Section 1.1: o Supports Secure Shell (SSH). o Supports Broker Security Enhancement (BSE). o TContextorControl component. • Added Section 2.1. • Updated Section 3.1.1. • Updated Section 3.2.1. • Deleted Section 3.2.2, “Edit Broker Servers Program,” because this application does not function on Windows 7 due to added security. An alternative is still being developed. • Updated Section 4.1. • Updated Section 4.2.1.2. • Added Section 4.2.4. • Updated Figure 2. **RPC Broker 1.1**	
07/25/2013	5.0	Tech Edit: • Baselined document. • Updated all styles and formatting to follow current internal team style template. • Updated all organizational references. **RPC Broker 1.1**	• Developer: H. W. • Technical Writer: T. B.

Date	Revision	Description	Authors
07/06/10	3.2	Updates for RPC Broker Patch XWB*1.1*50 (client-side only patch): - Added support for SSH for Attachmate Reflections (see Section 3.1.1). - Wrapped CCOW User Context into the primary TRPCBroker component and deleting the TCCOWRPCBroker component (see Section 4.2.1.2). - Support for Delphi 5.0, 6.0, 7.0, 2005, 2006, 2007, 2008, 2009, and 2010. - Changed references form Patch 47 to Patch 50 where appropriate. - Reformatted document to follow current PD National Documentation Standards and current style guidelines. **RPC Broker 1.1**	RPC Broker Development Team Oakland Office of Information and Technology Field Office (OI&TFO)
07/03/08	3.1	Updates for RPC Broker Patch XWB*1.1*47: - No content changes required; no new public classes, methods, or properties added to those available in XWB*1.1*40. - Bug fixes to the ValidAppHandle function and fixed memory leaks. - Support added for Delphi 2005, 2006, and 2007. - Reformatted document. - Changed references form Patch 40 to Patch 47 where appropriate. **RPC Broker 1.1**	- Oakland OI&TFO Development Team - Technical Writer: T. B.
02/22/05	3.0	Revised Version for RPC Broker Patch XWB*1.1*40 and previous undocumented patch updates. **RPC Broker 1.1**	- Oakland OI&TFO Development Team - Technical Writer: T. B.
02/19/02	2.0	Revised Version for RPC Broker Patch XWB*1.1*13. **RPC Broker 1.1**	- Oakland OI&TFO Development Team - Technical Writer: T. B.
09/97	1.0	Initial RPC Broker Version 1.1 software release. **RPC Broker 1.1**	- Oakland OI&TFO - Technical Writer: T. B.

Patch Revisions

For the current patch history related to this software, see the Patch Module on FORUM.

Table of Contents

1 Introduction

The Veterans Health Information Systems and Technology Architecture (VistA) Remote Procedure Call (RPC) Broker (also referred to as “Broker”) 1.1; RPC Broker Patch XWB*1.1*65 is now available.

RPC Broker 1.1 (fully patched) provides programmers with the capability to develop VistA client/server software. RPC Broker 1.1 also includes the Broker Development Kit (BDK), which provides updated components, properties, methods, and types. The BDK provides VistA application developers with the following features:

- Capability to create and implement client/server technology in the 32-bit Microsoft® Windows environment using the Broker component (e.g., create Delphi-based client/server VistA applications with Graphical User Interfaces [GUI]).
- Support for Commercial Off-the-Shelf (COTS) and Hybrid Open System Technology (HOST) client/server software using the Broker Dynamic Link Library (DLL).

RPC Broker 1.1 includes the following RPC Broker Delphi components for the 32-bit environment (listed alphabetically):

- TCCOWRPCBroker
- TContextorControl
- TRPCBroker
- TXWBRichEdit
- TXWBSSOiToken

NOTE: These RPC Broker components wrap the functionality of the Broker resulting in a more modularized and orderly interface. Those components derived from the original TRPCBroker component, inherit the TRPCBroker properties and methods.

2 Purpose

These release notes cover the latest changes to *RPC Broker 1.1; through Patch XWB*1.1*65.*

3 Audience

This document targets developers, system administrators, and users of *RPC Broker 1.1* and applies to the changes made between this release and any previous release for this software.

4 This Release

4.1 New Features and Functions Added

RPC Broker 1.1 client/server interface provides the following features and enhancements:

4.1.1 2-Factor Authentication Support

RPC Broker 1.1 TRPCBroker component enables 2-factor user authentication using the Secure Token Service (STS) delegated authentication model. 2-factor authentication (2FA) support on the VistA server is handled by VistA Kernel software. No RPC Broker changes are needed in VistA to enable 2-factor authentication on a Broker listener. Delphi RPC Broker client applications are automatically upgraded to the new 2-factor authentication method when using the current Broker Development Kit (BDK). (XWB*1.1*65)

A call is made to the Identity and Access Management (IAM) STS server, where a user is prompted for credentials. The current IAM requirement is for a Personal Identity Verification (PIV) card, which contains a Public Key Infrastructure (PKI) certificate assigned to the user. The PKI certificate is unlocked using a Personal Identification Number (PIN), satisfying the 2-factor authentication requirement. These user credentials are exchanged by IAM for a digitally signed Security Assertion Markup Language (SAML) token, which is passed to the VistA M Server. VistA validates the digital signature and integrity of the token, and uses attributes within the token to identify the authenticated user for access to the VistA M Server. RPC Broker client applications use the new Kernel XUS ESSO VALIDATE RPC to authenticate with a SAML token instead of the old XUS AV CODE RPC, which authenticated using Access/Verify codes.

NOTE: In the future, IAM may implement other authentication methods that will be exchanged for the same form of SAML token, so that no modification of RPC Broker or VistA will be required to implement the new authentication method.

4.1.2 IPv4/IPv6 Dual-Stack Environment Support

RPC Broker 1.1 supports IPv4/IPv6 Dual-Stack Environment. It upgraded Microsoft® Windows Application Programming Interfaces (APIs) from WinSock 1.1 IPv4 to WinSock 2.2 IPv4/IPv6 dual-stack. Applications compiled with the latest BDK will be protocol independent and able to connect to both IPv4 and IPv6 VistA servers. IPv4/IPv6 dual-stack support on the VistA server is handled by VistA Kernel software. No RPC Broker changes are needed in VistA to enable IPv6 on a Broker listener. (XWB*1.1*60)

IPv6 is a protocol designed to handle the growth rate of the Internet and to cope with the demanding requirements of services, mobility, and end-to-end security.

REF: A Federal Chief Information Office (CIO) "Transition to IPv6" memo released in September of 2010 requires agencies to continue their IPv6 transition efforts and has established specific milestones associated with enabling an IPv6 operational capability by the end of FY2014.

4.1.3 Secure Shell (SSH) Tunneling Support

RPC Broker 1.1 supports Secure Shell (SSH) tunneling using the highest level of encryption mutually available on both client and server. Data integrity message authentication codes (MACs) ensure that data is not altered in transit. Digital signatures are used for public key authentication to confirm that the party being authenticated (client application) holds the correct private key. SSH satisfies a need expressed by the Veterans Health Administration (VHA) information systems user community to provide secure data transfer between the client and the VistA M Server. (XWB*1.1*50)

The following data encryption standards are supported:

- Arcfour, Arcfour128, and Arcfour258 (stream mode)
- TripleDES (168-bit) CBC mode
- Cast (128-bit)
- Blowfish (128-bit) CBC mode
- AES (128-, 192-, or 256-bit) CBC mode and CTR mode

The following data integrity MAC standards are supported:

- hmac-sha1
- hmac-md5
- hmac-sha1-96
- hmac-md5-96
- hmac-ripemd-160
- hmac-sha256
- hmac-sha2-256
- hmac-sha512
- hmac-sha2-512

The following digital signature algorithms are supported:

- x509v3-rsa2048-sha256
- x509v3-sign-rsa
- x509v3-sign-dss
- ssh-rsa-sha2-256@attachmate.com
- ssh-rsa
- ssh-dss

Support is provided for:

- Micro Focus® Reflection—Terminal emulator software using SSH tunneling for clients within the VA to provide secure data transfer between the client and the VistA M Server.
- PuTTY Link (Plink)—Secure channels for clients using VistA outside of the VA.

4.1.3.1 Micro Focus® Reflection

For SSH tunneling using Micro Focus® Reflection, “SSH” is set as a command line option or as a property within the application (set to Micro Focus® Reflection). SSH is set to true if either of the following command line parameters are set:

- SSHPort=portnumber (to specify a particular port number – if not specified, it will use the port number for the remote server)
- SSHUser=username (for the remote server, where username is of the form ***xxxvista***, where the ***xxx*** is the station’s three letter abbreviation)

4.1.3.2 PuTTY Link (Plink)

For SSH tunneling using Plink.exe, “PLINK” is set as a command line option or as a property within the application (set to Plink). SSH is set to **true** if the following command line parameter is set:

SSHpw=password

4.1.4 Single Signon/User Context (SSO/UC) Support

RPC Broker 1.1 supports single sign-on (SSO) service with interfaces to VistA and *non*-VistA systems by using the TCCOWRPCBroker component. This was a need expressed by the Veterans Health Administration (VHA) information systems user community. This allows users to authenticate and sign on to multiple applications that are CCOW-enabled and Single Signon/User Context (SSO/UC)-aware using a single set of credentials, which reduces the need for multiple ID’s and passwords in the VistA clinician desktop environment. (XWB*1.1*40)

The TCCOWRPCBroker component allows VistA application developers to make their applications CCOW-enabled and SSO/UC-aware with all of the client/server-related functionality in one integrated component. Using the TCCOWRPCBroker component, an application can share User Context stored in the CCOW Context Vault.

Thus, when a VistA CCOW-enabled application is recompiled with the TCCOWRPCBroker component and other required code modifications are made, that application would then become SSO/UC-aware and capable of single sign-on (SSO).

REF: For more information on SSO/UC, see the *Single Sign-On/User Context (SSO/UC) Installation Guide* and *Single Sign-On/User Context (SSO/UC) Deployment Guide* on the VA Software Document Library (VDL) at: https://www.va.gov/vdl/application.asp?appid=162.

4.1.4.1 Disabling SSO/UC

For sites whose policy is *not* to allow the kinds of SSO-based logins supported by SSO/UC, the User Context-based SSO can be disabled by doing either of the following:

- Mark the User subject as “unshared” in the Sentillion Vergence Context Vault so that the User subject instance is kept separate for all application instances. This is how the Sentillion Vergence Context Vaults were initially configured when Veterans Health Administration (VHA) first procured them for Patient Context (i.e., User Context was specifically disabled).
- **Do *not* grant secure access in the Sentillion Vergence Context Vault to the application passcode used by the login components.** Without the application passcode, the login components *cannot* establish a secure binding to the User Context. This failure triggers a standard, non-SSO login process:
    1. The login component does not find a User Context.
    2. The login component prompts the user for their Access and Verify code credentials.
    3. The application logs in; and no User Context is set.

4.1.4.2 Kernel CCOW Login Token Expiration

The Kernel CCOW login token is valid from a minimum of 600 seconds to a maximum of 28,800 seconds (i.e., 10 minutes to 8 hours) from when the user first authenticated via Kernel on the VistA M Server. The default value is 5,400 seconds (i.e., 1.5 hours). This default value is a compromise between wanting to provide as rapid a Kernel CCOW login token expiration as possible for security reasons, versus the need for a SSO session to last long enough in order to be useful to the user.

To change the expiration time, system administrators can change the value stored in the CCOW TOKEN TIMEOUT field (#30.1) in the KERNEL SYSTEM PARAMETERS file (#8989.3).

4.1.5 Silent Logon Support

RPC Broker 1.1 provides “Silent Login” capability. It provides functionality associated with the ability to make logins to a VistA M Server without the RPC Broker asking for Access and Verify code information. Control of the Silent Logon functionality is maintained and administered on the server for both VistA client/server applications (i.e., GUI) and the roll-and-scroll environment (i.e., terminal sessions). (XWB*1.1*13)

The BDK provides two types of Silent Login:

- **Access/Verify Code-based—**Uses Access and Verify codes provided by the application. This type of Silent Login may be necessary for an application that runs as a background task and repeatedly signs on for short periods. Another case would be for applications that are interactive with the user, but are running under conditions where they cannot provide a standard dialogue window, such as that used by the Broker to request Access and Verify codes. Examples might be applications running on handheld devices or within a browser window.
- **Token-based—**Uses a token obtained by one application that is passed along with other information as a command line argument to a second application that it is starting. The token is obtained from the VistA server and remains valid for about twenty (20) seconds. When the newly started application sends this token during login the server identifies the same user and completes the login.

Due to the various conditions under which Silent Logins might be used, it was also necessary to provide options to the applications on error handling and processing. Applications that run as system services will crash if they attempt to show a dialogue box. Similarly, applications running within Web browsers are not permitted to show a dialogue box or to accept windows messages. Properties have been provided to permit the application to handle errors in a number of ways.

As a part of the Silent Login functionality, the TVistaUser class providing basic user information was added. This class is used as a property by the TRPCBroker class and is filled with data following completion of the login process. This property and its associated data are available to all applications, whether they are using a Silent Login or not.

4.1.6 32-Bit Processing and Delphi Support

RPC Broker 1.1 operates in a 32-bit Microsoft® Windows environment (i.e., client workstations running Microsoft® Windows 7, 8.1, or 10 operating systems). All RPC Broker components are upgraded to operate in a Microsoft® Windows 32-bit environment. (XWB*1.1*47)

NOTE: The current version does *not* support development in a 64-bit environment.

RPC Broker 1.1 supports Delphi 10 Berlin (10.1), 10 Seattle (10.0), XE8, XE7, XE6, XE5, and XE4. (XWB*1.1*65)

4.1.7 Broker Security Enhancement (BSE)

The RPC Broker 1.1 TRPCBroker component enables visitor access to remote sites using authentication established at a home site. (XWB*1.1*45)

4.1.8 *Non*-Callback Connections

By default the RPC Broker components are built with a UCX or *non*-callback Broker connection, so that it can be used from behind firewalls, routers, etc. (XWB*1.1*35)

4.1.9 Deferred RPCs

In order to increase efficiency, applications can run RPCs in the background.

4.1.10 Remote RPCs

In order to work with patient data across sites, applications can run RPCs on a remote server.

4.1.11 Multi-Instances Support

RPC Broker 1.1 supports multi-instances of the RPC Broker. RPC Broker code permits an application to open two separate Broker instances with the same Server/ListenerPort combination, resulting in two separate partitions on the server. Previously, an attempt to open a second Broker instance ended up using the same partition. For this capability to be useful for concurrent processing, an application would have to use threads to handle the separate Broker sessions. (XWB*1.1*13)

CAUTION: Although it is believed that there should be no problems, the RPC Broker is *not* guaranteed to be thread safe.

4.1.12 RPC Broker Components

RPC Broker 1.1 (fully patched) provides programmers with the capability to develop VistA client/server software using the following RPC Broker Delphi components in the 32-bit environment (listed alphabetically):

- TCCOWRPCBroker
- TContextorControl
- TRPCBroker
- TXWBRichEdit
- TXWBSSOiToken

NOTE: These RPC Broker components wrap the functionality of the Broker resulting in a more modularized and orderly interface. Those components derived from the original TRPCBroker component, inherit the TRPCBroker properties and methods.

REF: For a complete description of the RPC Broker components, properties, and methods, see the BDK Online Help (i.e., Broker_1_1.chm) or *RPC Broker Developer's Guide*.

4.1.12.1 TCCOWRPCBroker

4.1.12.1.1 TCCOWRPCBroker Component Added

The TCCOWRPCBroker component allows VistA application developers to make their applications CCOW-enabled and Single Sign-On/User Context (SSO/UC)-aware with all of the client/server-related functionality in one integrated component. Using the TCCOWRPCBroker component, an application can share User Context stored in the CCOW Context Vault. (XWB*1.1*40)

Thus, when a VistA CCOW-enabled application is recompiled with the TCCOWRPCBroker component and other required code modifications are made, that application would then become SSO/UC-aware and capable of single sign-on (SSO).

4.1.12.1.2 CCOW User Context Wrapped into the Primary TRPCBroker Component

RPC Broker 1.1 wraps CCOW User Context into the primary TRPCBroker component, so that if the Contextor property is set, then CCOW User Context will be used. This means that there is no need to have a separate TCCOWRPCBroker component. (XWB*1.1*50)

NOTE: All of the functionality used by and for the TCCOWRPCBroker component is still present, but it is now part of the regular TRPCBroker component.

4.1.12.2 TContextorControl

The TContextorControl component communicates with the Vergence Locator service. (XWB*1.1*40)

4.1.12.3 TRPCBroker

The original TRPCBroker Delphi component provides Delphi developers with an easy, object-based access to the Broker. It is compatible with the Delphi object oriented (OO) environment. This component, when placed on a Delphi form, allows you to connect to the server and reference M data within Delphi's Integrated Development Environment (IDE). It makes a Delphi form and everything on it "data aware."

4.1.12.4 TXWBRichEdit

The TXWBRichEdit component replaces the Introductory Text Memo component on the Login Form. TXWBRichEdit is a version of the TRichEdit component that uses Version 2 of Microsoft's® RichEdit Control and adds the ability to detect and respond to a Uniform Resource Locator (URL) in the text. This component permits you to provide some requested functionality on the login form. As an XWB namespaced component you are required to put it on the **Kernel** tab of the component palette, however, it rightly belongs on the **Win32** tab. (XWB*1.1*13)

4.1.12.5 TXWBSSOiToken

The TXWBSSOiToken component is made available separately from the TRPCBroker component for those Delphi applications that may need to do 2-factor authentication (2FA), but do *not* have a requirement to connect to a VistA M Server. For Delphi applications that connect to a VistA M Server, this functionality is included within the TRPCBroker component and there is no need to have a separate TXWBSSOiToken component added to the application. (XWB*1.1*65)

4.1.13 Classes Added

The following Classes were added to the RPC Broker 1.1:

- TVistaLogin (XWB*1.1*13)
- TVistaUser (XWB*1.1*13)
- TXWBWinsock (XWB*1.1* 40 and XWB*1.1*60)
- TXWBSSOiToken (XWB*1.1*65)

4.1.14 Library Methods Added

The following library methods were added to the TVCEdit Unit (XWB*1.1*13):

- ChangeVerify:
  ```
  function ChangeVerify(RPCBroker: TRPCBroker): Boolean;
  ```
- SilentChangeVerify:
  ```
  function SilentChangeVerify(RPCBroker: TRPCBroker; OldVerify, NewVerify1,
  NewVerify2: String; var Reason: String): Boolean;
  ```
- StartProgSLogin:
  ```
  procedure StartProgSLogin(const ProgLine: String; ConnectedBroker:
  TRPCBroker);
  ```

The following library methods were added to the TCCOWRPCBroker component (XWB*1.1*40):

- GetCCOWtoken:
  ```
  function GetCCOWtoken(Contextor: TContextorControl): string;
  ```
- IsUserCleared:
  ```
  function IsUserCleared: Boolean;
  ```
- IsUserContextPending:
  ```
  function IsUserContextPending(aContextItemCollection:
  IContextItemCollection):  Boolean;
  ```
- WasUserDefined:
  ```
  function WasUserDefined: Boolean;
  ```

4.1.15 Properties Added

The following properties were added to the RPC Broker 1.1:

4.1.15.1 TCCOWRPCBroker Properties

The following properties were added (XWB*1.1*40):

- CCOWLogonIDName (Public)
- CCOWLogonIDValue (Public)
- CCOWLogonName (Public)
- CCOWLogonNameValue (Public)
- CCOWLogonVpid (Public)
- CCOWLogonVpidValue (Public)
- Contextor (Public)

4.1.15.2 TRPCBroker Properties

The following properties were added (XWB*1.1*13, 35, 50, and 65):

- CCOWLogonIDName (Public)
- CCOWLogonIDValue (Public)
- CCOWLogonName (Public)
- CCOWLogonNameValue (Public)
- CCOWLogonVpid (Public)
- CCOWLogonVpidValue (Public)
- Contextor (Public)
- BrokerVersion (Public)
- CurrentContext (Public)
- KerneLogIn (Published)
- LogIn (Public)
- OnRPCBFailure (Public)
- RPCBError (Public)
- ShowErrorMsgs (Published)
- User (Public)
- SSHport (Public)
- SSHUser (Public)
- SSHpw (Public)
- SSOiToken (Public)
- SSOiSECID (Public)

- SSOiADUPN (Public)
- SSOiLogonName (Public)

4.1.15.3 TSharedBroker and TSharedRPCBroker Properties

The following Shared Broker properties were removed. The Shared Broker has been deprecated. (XWB*1.1*60)

- AllowShared (Public)
- OnConnectionDropped (Public)
- OnLogout(Published)

4.1.15.4 TVistaLogin Properties

The following properties were added (XWB*1.1*40):

- DomainName (Public)
- IsProductionAccount (Public)

4.1.15.5 TVistaUser Property

The following property was added (XWB*1.1*40):

- Vpid (Public)

4.1.15.6 TXWBSSOiToken Properties

The following properties were added (XWB*1.1*65):

- SSOiToken (Published)
- SSOiADUPN (Published)
- SSOiLogonName (Published)
- SSOiSECID (Published)

4.1.16 Types Added/Modified

The following Types were added to or modified in RPC Broker 1.1 (XWB*1.1*13 and XWB*1.1*40):

- TLoginMode
- TShowErorMsgs
- TOnLoginFailure
- TOnRPCBFailure
- TParamType

4.1.17 Separate Design-time and Run-time Packages

The BDK contains separate run-time and design-time packages. (XWB*1.1*14)

4.1.18 Source Code Availability

The BDK contains the Broker source code. The source code is located in the following directory:

BDK32\Source

CAUTION: Modified BDK source code should *not* be used to create VistA GUI applications.

Not all methods and properties found in the source code are documented at this time. Only those documented methods and properties are guaranteed to be made backwards compatible in future versions of the BDK.

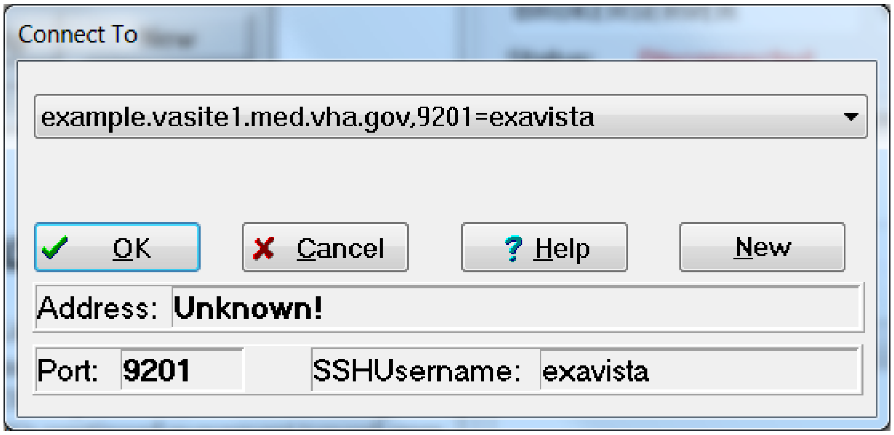
4.2 Enhancements and Modifications to Existing

4.2.1 GetServerInfo Function Modified

The GetServerInfo function obtains the end-user's target server and port. Use this function to set the TRPCBroker component's Server and ListenerPort properties before connecting to the server.

If there is more than one server/port to choose from, GetServerInfo displays an application window that allows users to select a service to connect to:

Figure 1: "Connect To" dialogue



example.vasite1.med.vha.gov,9201=exavista

Address: **Unknown!**

Port: **9201** SSHUsername: exavista

4.2.2 Dynamic Link Library (DLL) Interface Updated

RPC Broker 1.1 provides Dynamic Link Library (DLL) functions that allow applications written in *any* Microsoft® Windows-based development environment (e.g., Embarcadero's Delphi, Embarcadero C++, Microsoft® Visual Basic, and other COTS products), to take advantage of all the features offered by the RPC Broker component. This reflects VistA's continued movement toward open systems that support multiple GUI and client front-ends.

The Dynamic Link Library (DLL) functions act like a "shell" around the Delphi TRPCBroker component and provide developers with an easy function-based access to the Broker component. These functions allow GUI and client front-end applications written in Embarcadero's Delphi and other COTS products to take advantage of all the features that the Broker offers. All of the communication to the server is handled by the TRPCBroker component accessed via the DLL interface.

NOTE: The BAPI32.DLL contains all of the 32-bit Broker DLL functions. It provides an interface to the Broker component.

4.2.3 Library Methods Modified

The following library methods were modified (XWB*1.1*13):

- CheckCmdLine:

  ```
  function CheckCmdLine(SLBroker: TRPCBroker): Boolean;
  ```

  This was changed from a procedure to a function with a Boolean return value.

- GetServerInfo:

  The GetServerInfo library function in the RpcConf1unit, which can be used to select the desired Server name and ListenerPort, was modified to add a “new” button. This button can be used to add a new Server/ListenerPort combination to those available for selection. It will also accept and store a valid IP address, if no name is known for the location. This will permit those who have access to other Server/ListenerPort combinations that may not be available in the list on the current workstation to access them. However, they will still need a valid Access and Verify code to log on to the added location. Patch XWB*1.1*60 added a third field to store the SSHUsername for Secure Shell (SSH) connections. In other words, the Server/ListenerPort/SSHUsername combination is now stored in the Windows Registry for known VistA servers.

- TParams:

  The procedure Clear was moved from Private to Public.

- TRPCB Unit:

  ```
  TOnLoginFailure = procedure (VistaLogin: TVistaLogin) of object;
  Changed from Object: TObject, since this is what should be expected by the
  procedure if it is called.
  TOnRPCBFailure = procedure (RPCBroker: TRPCBroker) of object;
  ```

  Changed from Object: TObject, since this is what should be expected by the procedure if it is called.

4.3 Changes to the User Authentication Process—Guide for Technical Writers

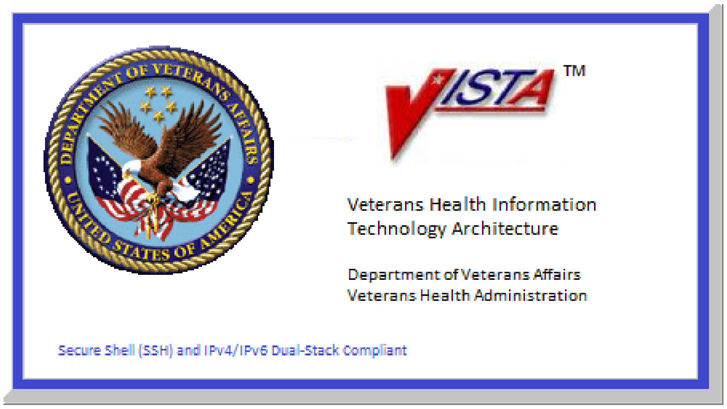
Delphi client applications compiled with the RPC Broker Patch XWB*1.1*65 Broker Development Kit (BDK) will implement user authentication, identification, and authorization features that are changed in this release. The following information is provided as a guide to technical writers documenting these changes in Delhi client application documentation.

4.3.1 Validation of Users

The VistA Sign-on dialogue is invoked when the client application connects to the VistA server.

After starting the application, many applications display a splash screen. An example of a VistA application splash screen is shown in Figure 2:

Figure 2: Sample VistA Application "Sign-on" Splash Screen



Veterans Health Information
Technology Architecture

Department of Veterans Affairs
Veterans Health Administration

Secure Shell (SSH) and IPv4/IPv6 Dual-Stack Compliant

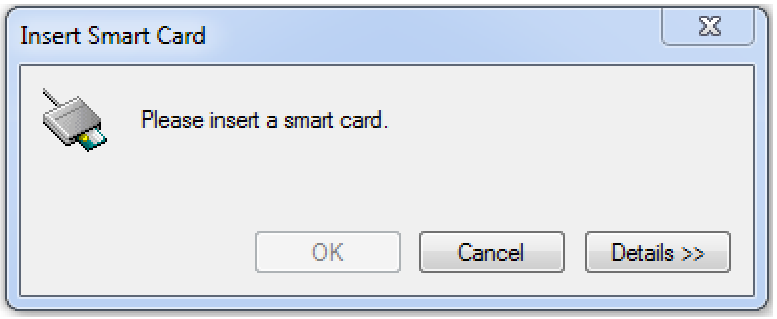
4.3.1.1 VistA 2-Factor Authentication Dialogue

When the client application opens, the user is prompted for 2-factor authentication (2FA) as the preferred form of user authentication.

An example of 2-factor authentication (2FA) follows:

1. If a user does *not* have a PIV Smart Card inserted, the system prompts them as shown in Figure 3. Selecting "Cancel" will fail over to Access and Verify code authentication.

Figure 3: Microsoft Windows Security: PIV Smart Card Prompt



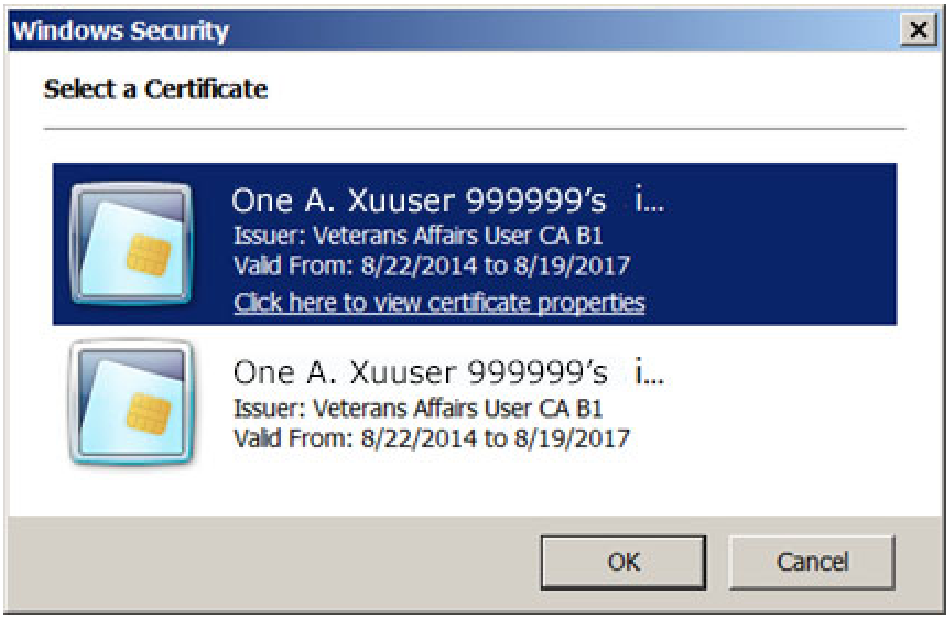
Insert Smart Card

Please insert a smart card.

2. After inserting a PIV Smart Card, the system displays the available Public Key Infrastructure (PKI) certificates from which to choose, as shown in Figure 4.

   Selecting **Cancel** at this point fails over to Access and Verify code authentication.

Figure 4: Microsoft Windows Security: PKI Certificate Selection Dialogue



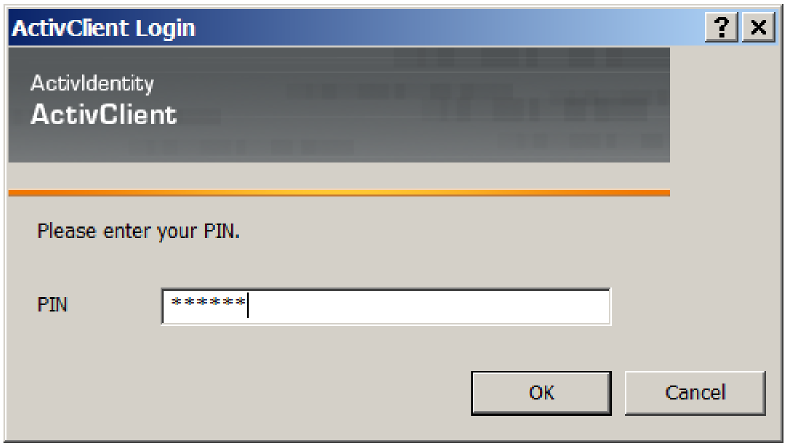
One A. Xuuser 999999's i...
Issuer: Veterans Affairs User CA B1
Valid From: 8/22/2014 to 8/19/2017
Click here to view certificate properties

One A. Xuuser 999999's i...
Issuer: Veterans Affairs User CA B1
Valid From: 8/22/2014 to 8/19/2017

3. After selecting a valid certificate, the user is prompted to enter a Personal Identification Number (PIN), as shown in Figure 5.

   Selecting **Cancel** at this point fails over to Access and Verify code authentication.

Figure 5: ActivClient Login: PIN Dialogue



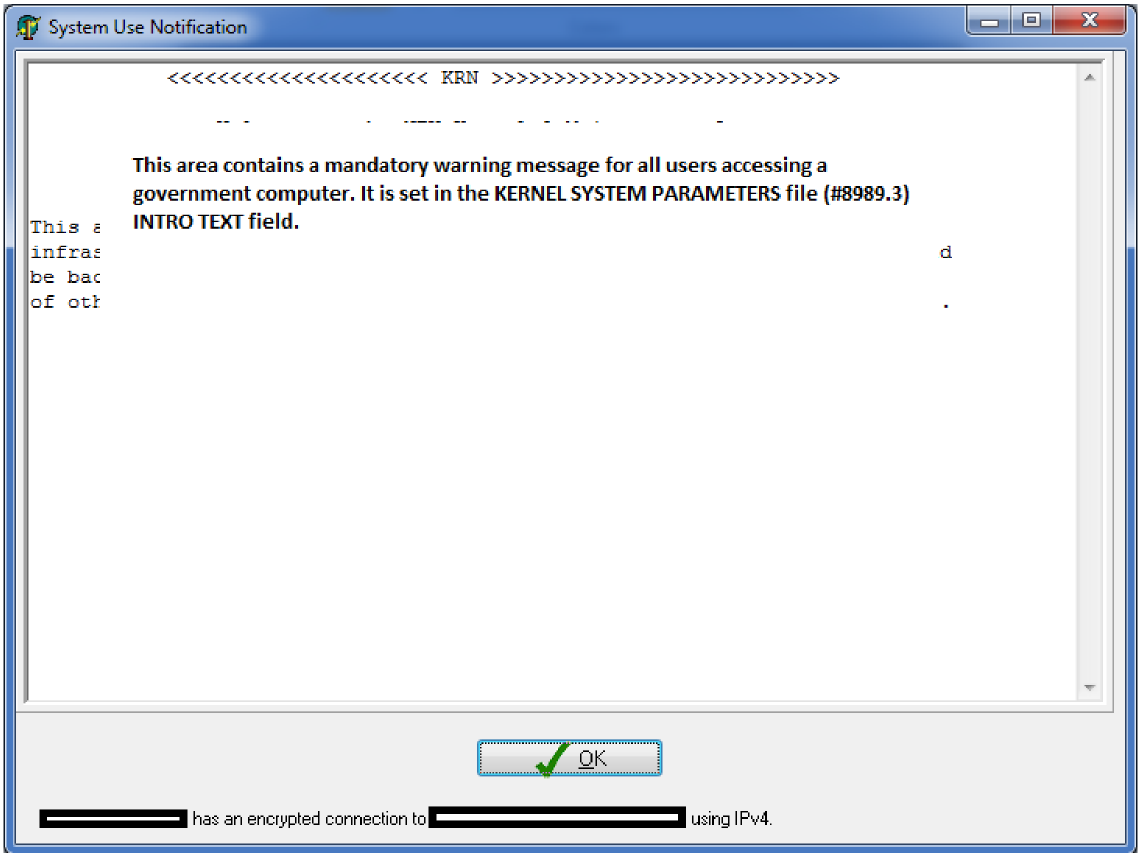
Please enter your PIN.

PIN ******

4. After entering a PIN, there is a short system delay as the user is authenticated and identified.

5. A mandatory “System Use Notification” warning message is then displayed to the user as shown in Figure 6.

Figure 6: Sample System Use Notification



```
            <<<<<<<<<<<<<<<<<<<<< KRN >>>>>>>>>>>>>>>>>>>>>>>>>>>>
```

This area contains a mandatory warning message for all users accessing a government computer. It is set in the KERNEL SYSTEM PARAMETERS file (#8989.3) INTRO TEXT field.

```
This a
infras                                                         d
be bac
of oth                                                         .
```

[redacted] has an encrypted connection to [redacted] using IPv4.

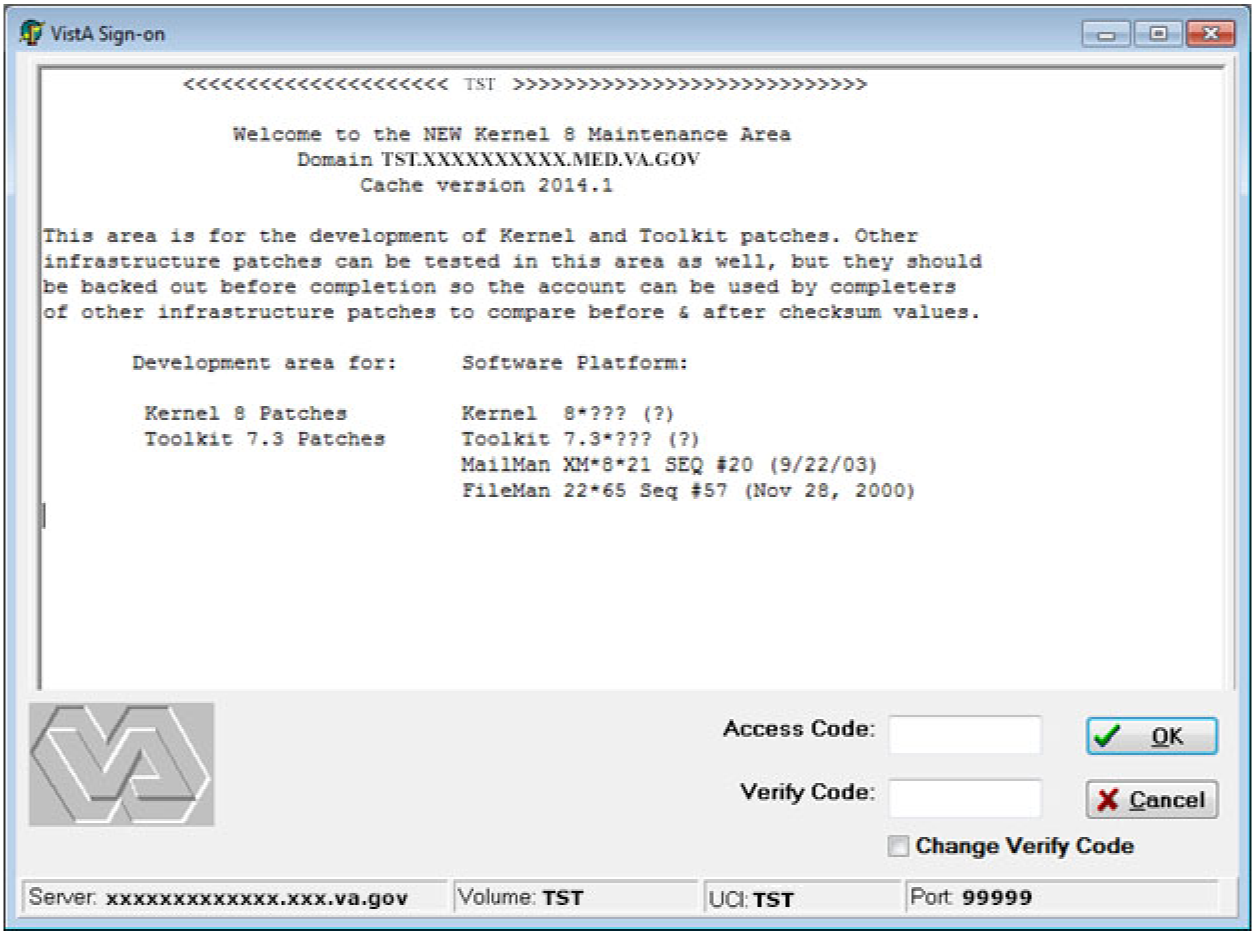
4.3.1.2 VistA Access/Verify Code Signon Dialogue

If 2-factor authentication (2FA) fails or is cancelled, the authentication process fails over to Access and Verify code authentication.

Selecting **Cancel** at this point displays an error and the user is disconnected.

A sample of the VistA Access and Verify code Sign-on dialogue is illustrated Figure 7:

Figure 7: Sample VistA Sign-on Security Dialogue: Access and Verify codes



```
             <<<<<<<<<<<<<<<<<<<<< TST >>>>>>>>>>>>>>>>>>>>>>>>>>>>

                  Welcome to the NEW Kernel 8 Maintenance Area
                       Domain TST.XXXXXXXXXXX.MED.VA.GOV
                            Cache version 2014.1

This area is for the development of Kernel and Toolkit patches. Other
infrastructure patches can be tested in this area as well, but they should
be backed out before completion so the account can be used by completers
of other infrastructure patches to compare before & after checksum values.

       Development area for:     Software Platform:

        Kernel 8 Patches         Kernel  8*??? (?)
        Toolkit 7.3 Patches      Toolkit 7.3*??? (?)
                                 MailMan XM*8*21 SEQ #20 (9/22/03)
                                 FileMan 22*65 Seq #57 (Nov 28, 2000)
```

Access Code:

Verify Code:

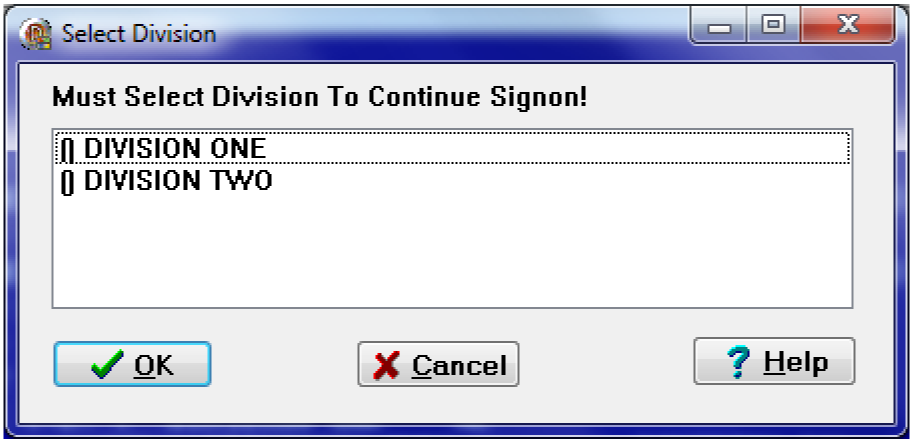
□ Change Verify Code

Server: xxxxxxxxxxxxx.xxx.va.gov | Volume: TST | UCI: TST | Port: 99999

4.3.1.3 VistA Division Selection Dialogue

After completing user authentication and identification, the process of user authorization continues. If a user is associated with more than one institution (division), the user is presented with a dialogue similar to Figure 8:

Figure 8: Select Division Dialogue—Sample Entries



Must Select Division To Continue Signon!

[] DIVISION ONE
[] DIVISION TWO

To continue the sign-on process, the user *must* select a division from the list presented. The user's default division is initially highlighted. To choose a different division, users should click on or use the arrow keys to highlight the appropriate division and press **OK** after making their selection.

Selecting **Cancel** at this point displays an error and the user is disconnected.

The final step of user authorization is usually transparent to the user. The user *must* be assigned the "context" menu option associated with the client application along with any security keys used to control access to the application. If the user does *not* have the required menu option and security keys assigned, then an error is displayed and the user is disconnected.

4.4 Known Issues

There are no known issues with RPC Broker 1.1.

5 Product Documentation

The following product documentation is available with RPC Broker 1.1.

5.1 RPC Broker Documentation

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

- *RPC Broker Release Notes* (this manual)
- *RPC Broker Deployment, Installation, Back-Out, and Rollback Guide*
- *RPC Broker Systems Management Guide*
- *RPC Broker User Guide*
- *RPC Broker Technical Manual*
- *RPC Broker Developer's Guide*—Document and RPC Broker BDK Online Help

- RPC Broker VA Intranet website.

  This site provides announcements, additional information (e.g., Frequently Asked Questions [FAQs], advisories), documentation links, archives of older documentation and software downloads.

VistA documentation is made available online in Microsoft® Word format and in Adobe Acrobat Portable Document Format (PDF). The PDF documents *must* be read using the Adobe Acrobat Reader, which is freely distributed by Adobe Systems Incorporated at: http://www.adobe.com/

VistA documentation can be downloaded from the VA Software Document Library (VDL) Website: http://www.va.gov/vdl/

The RPC Broker documentation is located on the VDL at: https://www.va.gov/vdl/application.asp?appid=23

VistA documentation and software can also be downloaded from the Product Support (PS) Anonymous Directories.

5.2 RPC Broker BDK Online Help

RPC Broker 1.1 BDK includes online help (i.e., Broker_1_1.chm), which provides an overview of development with the RPC Broker; it contains the same topics and information found in the *RPC Broker Developer's Guide*. The online help also includes other related topics for system administrators and Broker developers (e.g., Tutorials, RPC information, troubleshooting and debugging tips).

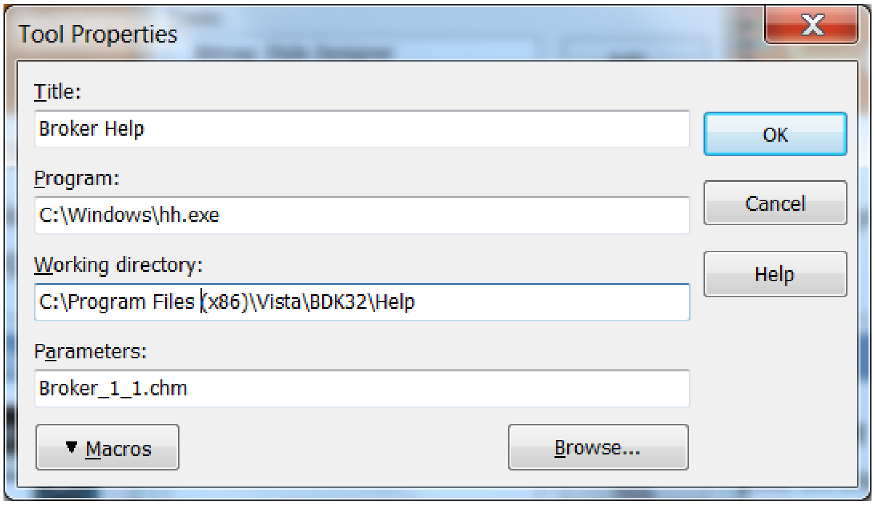
The BDK Online Help is distributed in two zip files:

- Broker_1_1.zip (i.e., Broker_1_1.chm)—This zip file contains the standalone online HTML help file. Unzip the contents and double-click on the **Broker_1_1.chm** file to open the help.
- Broker_1_1-HTML_Files.zip—This zip file contains the associated HTML help files. Unzip the contents in the same directory and double-click on the **index.htm** file to open the help.

You can access context-sensitive help (i.e., Broker_1_1.chm) for the RPC Broker-related components and associated DLL exported procedures and functions by selecting the component on a form, or highlighting a particular procedure or function, and pressing the **F1** key to get help on that item.

You can create an entry for Broker_1_1.chm in Delphi's Tools Menu, to make it easily accessible from within Delphi. To do this, use Delphi's **Tools | Configure Tools** option and create a new menu entry as shown in Figure 9.

Figure 9: Delphi’s Tool Properties dialogue—Broker_1_1.chm entry



Tool Properties

Title: Broker Help

Program: C:\Windows\hh.exe

Working directory: C:\Program Files (x86)\Vista\BDK32\Help

Parameters: Broker_1_1.chm