Sunrise Clinical Manager™ 5.0 XA™

Technical Reference Guide

P/N 214772 D



For further information about this manual or other Eclipsys Corporation products, contact Product Support Services, as follows:

Email: customer.support@eclipsys.com

Telephone: 888-GET-HELP (888-438-4357)

Fax: (602) 302-8555

International customers should call (602) 302-8500 for Product Support Services.

Technology Solutions Center fax: (973) 316-8587

Technology Solutions Center email: tsc.support@eclipsys.com.

Eclipsys welcomes your opinion regarding this document. Please email your comments and suggestions to techpubs@eclipsys.com or send them to the Product Support Services email address shown above.

Copyright © 2007, Eclipsys Corporation. All rights reserved.

Eclipsys is a registered trademark of Eclipsys Corporation.

All other products are trademarks or registered trademarks of their respective holders, all rights reserved. Reference to these products is not intended to imply affiliation with or sponsorship of Eclipsys Corporation.

Proprietary Notice

This document is an unpublished work containing confidential and trade secret information, which is proprietary to Eclipsys Corporation (“Eclipsys”), and is protected by laws pertaining to such materials. This document, the information in this document, and all rights thereto are the sole and exclusive property of Eclipsys, are intended for use by customers and employees of Eclipsys, and are not to be copied, used, or disclosed to anyone else, in whole or in part, without the express written permission of Eclipsys. For authorization from Eclipsys to copy this information, please call Eclipsys Product Support Services at 888 GET-HELP or 888 438-4357.

Table of Contents

Chapter 1: About This Guide 1
1.1 Overview 1
1.2 Audience 2
1.3 What's New in This Document 2
1.4 Accessing Eclipsys Application Online Help 2
1.5 Contact Us 2
1.6 Using Adobe® Reader® 3
Chapter 2: Automation Control Interface 5
2.1 Overview 5
2.1.1 Single Unified Login Shared Between Desktop Applications 5
2.1.2 Managing Logon Operations 6
2.1.3 Desktop Management for Hiding and Displaying Sunrise Clinical Manager 6
2.1.4 Desktop Management of SCM Tools Menu Items 7
2.1.5 HL7 Desktop Integration 7
2.2 Starting the Correct Version of SCM 8
2.3 The Interfaces 8
2.4 _Application 8
2.4.1 Application 10
2.4.2 Parent 10
2.4.3 Name 11
2.4.4 FullName 12
2.4.5 Visible 13
2.4.6 Quit 13
2.4.7 SessionID 14
2.4.8 ActiveTab 15
2.4.9 Logon 16
2.4.10 Logoff 17
2.4.11 ChangePassword 18
2.4.12 AddTrigger 18
2.4.13 RemoveTrigger 20
2.4.14 AddTool 21
2.4.15 RemoveTool 22
2.4.16 EnableUIPasswordChange 22
2.4.17 Application_OnToolExecute(nMenuID) 23
2.4.18 Application_OnUserExit( ) 24
2.4.19 Application_OnUserLogoff( ) 24
2.5 HL7 Desktop Integration Support 25
2.5.1 IHL7Desktop Interface 25
2.5.2 HL7DesktopState 26
2.5.3 JoinHL7Desktop 28

This page contains Eclipsys proprietary information and is not to be duplicated or disclosed to unauthorized persons.

2.5.4 LeaveHL7Desktop 29
2.5.5 AllowDataLossOnContextChange 30
Chapter 3: Context Server 31
3.1 Overview 31
3.2 Data Representation 32
3.2.1 Session Control 33
3.3 Context Server Installation and Configuration 33
3.3.1 Enabling the Context Server 34
3.3.2 Starting SCM from the Command Line 34
3.4 Referencing the Classes and Interfaces 35
3.5 The Context Server Interfaces 36
3.5.1 IHVCContextManager 36
3.5.2 GetCurrentContextToken 38
3.5.3 CreateContextSession 39
3.5.4 JoinContextSession 39
3.5.5 LeaveContextSession 41
3.5.6 GetApplicationSessions 42
3.5.7 GetApplicationSessionsAndPIDs 43
3.5.8 BeginContextChanges 44
3.5.9 PublishContextChanges 46
3.5.10 TakeOwnership 47
3.5.11 ReleaseOwnership 48
3.5.12 FreezeContextSession 49
3.5.13 ReleaseSession 51
3.5.14 IHVCContextData 52
3.5.15 GetContextItems 52
3.5.16 GetContextValues 53
3.5.17 SetContextValues 54
3.5.18 IHVCContextParticipant 56
3.5.19 Participant_ContextChanged(nContext) 56
3.5.20 Participant_Ping() 56
3.6 Objects 57
3.6.1 Class CoContextManager 57
3.6.2 Context Data 57
Chapter 4: Media Viewer 61
4.1 Overview 61
4.2 Configuring Media Display 62
4.2.1 Ensuring that the Server is Registered 62
4.2.2 Adding the Server to the Registered Media Viewers Dictionary 63
4.2.3 Confirm Media File Type Associations 68
4.2.4 Verifying the Location of the Image Database 68
4.2.5 Configuring Support for Sunrise PACS 69

This page contains Eclipsys proprietary information and is not to be duplicated or disclosed to unauthorized persons.

4.3 Using HL7 Messages . . . 70
4.4 Creating a Media Viewer Server . . . 72
4.4.1 Server Requirements . . . 72
4.4.2 IHVCMediaViewer Interface Specification . . . 73
Chapter 5: Desktop Integration in Sunrise Clinical Manager . . . 77
5.1 Overview . . . 77
5.2 Common Terms and Phrases . . . 78
5.3 Prerequisite Tasks . . . 81
5.4 Modes of Context Sharing . . . 81
5.4.1 Peer-to-Peer . . . 82
5.4.2 Master . . . 82
5.4.3 Setting the Participation Mode . . . 82
5.5 Configuring the Vergence Context Manager for View Context Support . . . 83
5.6 Configuring Sunrise Clinical Manager for Desktop Integration . . . 86
5.6.1 Configuring View Change Requests . . . 86
5.6.2 Setting the Tools Menu Options Using the Environment Profile . . . 86
5.6.3 Adding Selections to the Tools Menu in Sunrise Clinical Manager . . . 87
5.6.4 Configuring Tools Menu Actions . . . 90
5.6.5 Enabling a Shortcut Key . . . 91
5.6.6 Configuring Medical Record Numbers (MRNs) . . . 91
5.6.7 Configuring Exit of all Participant Applications . . . 94
5.6.8 Configuring Sunrise Clinical Manager Context Controller . . . 96
5.7 Managing the Clinical Link . . . 97
5.7.1 Joining the Context . . . 97
5.7.2 Viewing the Clinical Link State in Sunrise Clinical Manager . . . 97
5.7.3 Clinical Link Icons . . . 97
5.7.4 The Clinical Link Menu . . . 98
5.7.5 Rejoining the Context . . . 98
5.7.6 Setting the View . . . 99
5.7.7 Patient List Features for Clinical Link List . . . 99
5.8 Responding to Context Changes . . . 99
5.8.1 Responding to User Context Changes . . . 99
5.8.2 Responding to Patient Context Changes . . . 100
5.8.3 Changing the Context with Unsaved Data . . . 101
5.8.4 Changing the Context When Sunrise Clinical Manager is Minimized . . . 103
5.8.5 Changing the Context When Sunrise Clinical Manager is Suspended . . . 103
Chapter 6: Creating an ActiveX Custom Tab . . . 105
6.1 Overview . . . 105
6.2 The Shell Data Sink Interfaces . . . 106
6.2.1 IPageViewSink::GetPatientRecordNames . . . 107
6.2.2 IPageViewSink::GetPatientRecordValues . . . 109
6.3 The ActiveX Control Interfaces . . . 110

This page contains Eclipsys proprietary information and is not to be duplicated or disclosed to unauthorized persons.

6.3.1 IPageView::AdviseSink . . . 111
6.3.2 IPageView::UnadviseSink . . . 112
6.3.3 IPageView::OnChangeVisit . . . 113
6.3.4 IPageView::GetActionMenu . . . 114
6.3.5 IPageView::FireActionMenuItem . . . 115
6.3.6 IPageView::CanChangePatient . . . 116
6.3.7 IPageView::CanCloseTab . . . 118
6.3.8 IPageView::SetFocusFirst . . . 119
6.3.9 IPageView::SetFocusLast . . . 119
6.3.10 IPageView::IsFirstWnd . . . 120
6.3.11 IPageView::IsLastWnd . . . 121
6.4 Using C++ . . . 121
6.5 Getting Started With Visual Basic . . . 122
6.6 Accessing Sunrise Clinical Manager Data . . . 124
6.7 Controlling Tabbing Behavior . . . 126
6.7.1 Example . . . 126
6.8 Obtaining Data From the Sunrise Clinical Manager Database . . . 128
6.8.1 Security Considerations . . . 128
6.8.2 Connecting to the Database . . . 129
6.8.3 Obtaining Patient Data . . . 130
6.8.4 Checking Sunrise Clinical Manager User Security Rights . . . 135
6.9 Adding New ActiveX Tab Entries to Sunrise Clinical Manager Environment Profiles . . . 137

Chapter 7: Authentication options . . . 141
7.1 Overview . . . 141
7.2 Authentication Options . . . 141
7.2.1 Username/Password Authentication . . . 142
7.2.2 Biometric Authentication . . . 142
7.2.3 Identix Biometric Requirements . . . 143
7.2.4 Two-Factor Username/Password + Identix Biometric . . . 143
7.2.5 RSA SecurID . . . 144
7.3 Configuring Authentication . . . 145
7.3.1 Logon Authentication Options . . . 145
7.3.2 Related Environment Profiles: Authentication Option . . . 145
7.3.3 Re-Authentication Options . . . 146
7.3.4 Related Environment Profiles: Re-Authentication Option . . . 147
7.3.5 Re-authentication Trigger Events . . . 147
7.3.6 Workstation-Specific Authentication Options . . . 148
7.3.7 Alternate Authentication . . . 148
7.3.8 Related Environment Profiles: Alternate Authentication Options . . . 149
7.4 Authentication in a Citrix Environment . . . 150
7.5 Special Authentication Conditions . . . 151
7.5.1 Workstation Not Configured With Biometric Device . . . 151
7.5.2 Workstation Biometric Device Not Working . . . 152

This page contains Eclipsys proprietary information and is not to be duplicated or disclosed to unauthorized persons.

7.5.3 Remote User Authentication 152
Index 153

This page contains Eclipsys proprietary information and is not to be duplicated or disclosed to unauthorized persons.

This page contains Eclipsys proprietary information and is not to be duplicated or disclosed to unauthorized persons.

CHAPTER 1: ABOUT THIS GUIDE

Contents

1.1 Overview 1
1.2 Audience 2
1.3 What's New in This Document 2
1.4 Accessing Eclipsys Application Online Help 2
1.5 Contact Us 2
1.6 Using Adobe® Reader® 3

1.1 Overview

The Technical Reference guide provides additional technical information outside the scope of a Configuration guide. While SCM is a sophisticated clinical management application, it is also a Windows-based desktop application. Many desktop applications on Windows-based platforms provide interoperability.

SCM provides interoperability with other Windows-based applications to facilitate these features:

- Single unified login shared between desktop applications.
- Desktop management for hiding and displaying SCM.
- Desktop management of SCM Tools menu items.
- Control of tab selection within SCM.
- Patient List refreshment after third-party data updates.

Note: Some graphics of application windows and the names of some applications described in this guide may be slightly different than those shown on your system due to the highly configurable nature of Eclipsys software. These graphics are for illustration purposes only and these variances do not affect the functionality of the application. The software described in this guide is based on a core or model product and may not fully represent customer modifications.

This page contains Eclipsys proprietary information and is not to be duplicated or disclosed to unauthorized persons.

1.2 Audience

The intended audience for this guide is a developer or implementation team.

1.3 What's New in This Document

The following table defines changes to the document for the current release.

Table 1.1: What's New

Chapter	Description
Chapters 6, 7, 8, and 10 (from old Technical Reference guide)	Took out the following chapters to include in the *Sunrise Clinical Manager Objects Plus/XA Guide:* .NET Custom Tabs, Configuring Custom Applications, Configuring Custom Form Applications, and SCM Library Objects chapters.
All Chapters	Updated the guide to the new template.
Chapter 7	Authentication Options used to be Strong Authentication and has been updated.

1.4 Accessing Eclipsys Application Online Help

Eclipsys applications have online help systems that you can view to learn more about the part of the application you are using.

◆ To access online help

- On most windows, press F1.
- Click Help buttons in the applications to access context-sensitive help.
- Select Help from the toolbar. You can launch the main help, search the help, or view a page that identifies the current version of the application you are running.

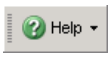
1.5 Contact Us

- If you have a problem with an Eclipsys product and need help, call Product Support at 1-888-GET-HELP or send an email to customersupport@eclipsys.com.
- If you have feedback about the documentation, contact us at userdoc@eclipsys.com.

This page contains Eclipsys proprietary information and is not to be duplicated or disclosed to unauthorized persons.

1.6 Using Adobe® Reader®

This document is provided in PDF format, which you view using Adobe® Reader® 7.0.

If you require help using Adobe Reader, press F1 or click 

on the toolbar.

To rotate a page, select View > Rotate View > Clockwise or Counterclockwise.

This page contains Eclipsys proprietary information and is not to be duplicated or disclosed to unauthorized persons.

This page contains Eclipsys proprietary information and is not to be duplicated or disclosed to unauthorized persons.

CHAPTER 2: AUTOMATION CONTROL INTERFACE

Contents

2.1 Overview 5
2.2 Starting the Correct Version of SCM 8
2.3 The Interfaces 8
2.4 _Application 8
2.5 HL7 Desktop Integration Support 25

2.1 Overview

Automation is a feature of the Component Object Model (COM), an industry-standard technology that applications use to expose their objects to development tools, macro languages, and other applications that support automation.

Because Sunrise Clinical Manager supports automation, the objects SCM exposes can be accessed by Visual Basic. Visual Basic is used to manipulate these objects by invoking methods on the object, or by getting and setting the object's properties. SCM's object model allows sites to integrate SCM into third-party or custom desktop management software. The details of the properties and methods supported by SCM, including sample Visual Basic code, are provided later in this chapter.

2.1.1 Single Unified Login Shared Between Desktop Applications

SCM has a complex initializing routine; code is loaded before the Logon window opens to reduce the time from the Logon window to the Patient List appearing. This delay increases the perceived performance of SCM. The automation controller startup, however, does not present a SCM Logon window to the user. The automation controller performs all the initialization in one step.

The SCM automation controller is a single-use controller. Each call for the SCM application object creates a new instance of SCM on the desktop. This is consistent with SDI architecture and controllers.

When the Application Management Module (AMM) starts, it initializes SCM before it calls the logon routines. When the AMM creates a context object as a

This page contains Eclipsys proprietary information and is not to be duplicated or disclosed to unauthorized persons.

result of a user logon, SCM is attached to the session and the non-interactive logon initiated. SCM does not appear until the user selects the Patient List button in the AMM.

These operations are identical in all applications controlling SCM. The controller initializes SCM at startup. When a context session is started, the controller attaches SCM to the session. When a user logs on, the controller calls the logon methods. Initialization always assumes SCM to be part of a context session. If the controller calls logon routines before setting a context session, SCM starts its own context session to which the controller can attach.

Unlike most automation controllers, SCM is fully operated by an end-user while another application holds the interface. This means that the user cannot close SCM and leave the launching application with an invalid interface. To the user, SCM gives the appearance of shutting down, but it is actually activating the controlling application and hiding SCM.

2.1.2 Managing Logon Operations

When SCM is running in stand alone mode, the logon procedure occurs in the middle of the initialization process. With an automation controller, the logon process is not part of initialization. SCM provides a way for the logon to be used in a non-interactive mode. The logon process has been developed so that it behaves consistently; regardless of whether this is the first or a subsequent logon process, because the interface always calls the same methods.

SCM is a stalled application when a user logs out. The application remains in the Logon window message pump while waiting for a user to log on to SCM. The logon methods create the same situation, because the SCM is unable to operate without a database connection, which requires an active user.

The solution has been to ensure that a message pump operates in the logout session, which allows the processing of messages for COM while keeping SCM in a semi-idle state without a database connection. SCM prevents the controlling application from making SCM visible to users while in this state. The only legal operations in this state are changing context sessions, logging on a new user, or terminating SCM.

Only the controlling application terminates SCM. SCM never allows a user to terminate the application. Otherwise, the application controlling SCM would have an invalid reference to SCM. This ensures that user passwords are kept consistent with those given to the Application Management Module (AMM).

2.1.3 Desktop Management for Hiding and Displaying Sunrise Clinical Manager

A visible property is a rarely used part of the automation controller standard, because most controllers work as a background process. Displaying the application allows users to interact with the application while another application holds the controller interface. This scenario is rare; because this shared interaction is fraught with problems. An application must not allow a user to shut down the

This page contains Eclipsys proprietary information and is not to be duplicated or disclosed to unauthorized persons.

running instance of the application because the user can invalidate the state of the controller interface held by the initiating application. Careful implementation ensures that the user cannot interfere with application control, and vice versa.

2.1.4 Desktop Management of SCM Tools Menu Items

The Tools menu connects event sinks to individual menu items to provide a callback into the application with the controller interface. When the user selects a menu item, SCM calls the controlling application, and identifies the user selection, and the controller application rather than SCM responds to the menu choice.

The AMM (or any controller) must register an event sink for each menu item. To register a menu item, the controller must provide a sink, a menu string, and a prompt string. The sink is called when the menu item is selected. The menu string is the literal text appearing in the menu. The prompt string is the text that appears in the Status bar when the user selects the menu item.

Note: Access an application defined by automation from an enterprise-defined column of the Patient List. Use the Double-Click Application options on the General tab of the New Visit List Column window. The menu item that automation added to the Tools menu does not appear in the Name drop-down list, as it does when adding an item to the Tools menu using the environment profile. Manually type the same string in the Name field that automation used to define the menu item on the Tools menu. For example, if automation used &Calculator as the string to access a calculator application, Enter &Calculator in the Name field. Patient List Refreshment After Third-Party Data Updates.

Because SCM can remain hidden for long stretches as a user works with other applications, SCM should be refreshed before viewing. An application controlling SCM sometimes needs to trigger a refresh because of database activity generated through HL7 interfaces. The application controller interface provides this functionality.

2.1.5 HL7 Desktop Integration

SCM can be configured to share clinical context information with other clinical applications. To have SCM behave in a master or peer role to another clinical application, use the SCM Context Controller application provided by Eclipsys, or create a custom automation controller using the information provided in this chapter.

To use the SCM Context Controller, see "Desktop Integration in Sunrise Clinical Manager" on page 77. To create your own automation controller, see "HL7 Desktop Integration Support" on page 25."

For information on having SCM operate as a peer to another application, see "Desktop Integration in Sunrise Clinical Manager" on page 77.

This page contains Eclipsys proprietary information and is not to be duplicated or disclosed to unauthorized persons.

2.2 Starting the Correct Version of SCM

When the SCM client (hvc.exe) is installed on a workstation, a generic (non-version specific) registry key is created. There is a single ProgID in the registry (CareVISION.Application) that maps to a folder where hvc.exe is found. Every time the SCM client is installed, the location of this folder is overwritten. If more than one version of SCM is installed on a workstation, the automation controller interface always launches the most recently installed version of SCM, regardless of the version that has been specified through the SCM Assistant.

If the version of SCM you want the automation controller to launch is not the most recently installed version, edit the registry key.

◆ To edit the registry key to start the correct version of SCM

1 From the Start menu, select Run.

2 In the Open field, type regedit (or regedit32 if applicable), and click OK.

3 Navigate to `HKEY_CLASSES_ROOT\CLSID\{06A75124-BA96-11d1-B8AD-006097CCF9F1}\LocalServer32`.

4 Right-click the application path key and select Modify.

5 Change the full path to the version of SCM that you want the automation controller to launch, as shown in the example below:

```
C:\Program Files\Eclipsys Sunrise\Clinical Manager
Client\Version\hvc.exe
```

2.3 The Interfaces

This section presents the _Application interface and its associated methods and properties.

2.4 _Application

T_Application is the standard application interface defined in the Automation Programmer's Reference. In addition, methods and properties specifically identified above have been added to the standard default methods and properties.

```
interface _Application : IDispatch
{
// Standard automation properties
[propget]
HRESULT Application([out, retval]
Application** ppApplication);
[propget]
HRESULT Parent([out, retval] IDispatch** ppParent);
[propget]
HRESULT Name([out, retval] BSTR* pbstrName);
```


This page contains Eclipsys proprietary information and is not to be duplicated or disclosed to unauthorized persons.

```
[propget]
HRESULT FullName([out, retval] BSTR* pbstrFullName);
[propget]
HRESULT Visible([out, retval] BOOL* pbVisible);
[propput]
HRESULT Visible([in] BOOL bVisible);

// Standard Automation methods
HRESULT Quit();

// Additional CV specific properties
[propget]
HRESULT SessionID([out, retval] long* pnSessionID);
[propput]
HRESULT SessionID([in] long nSessionID);
[propget]
HRESULT ActiveTab([out,retval] long nTab);
[propput]
HRESULT ActiveTab([in] long nTab);

// Additional CV specific methods
HRESULT Logon([in] BSTR bstrUserName, [in] BSTR bstrPassword);
HRESULT Logoff();
HRESULT AddTrigger([in] LPDISPATCH pDispatch,
 [out,retval] long* nCookie);
HRESULT RemoveTrigger([in] long nCookie);
HRESULT ChangePassword([in] BSTR bstrOldPassword,
 [in] BSTR bstrNewPassword);

HRESULT RefreshPatientList();

// Tools callbacks
HRESULT AddTool([in] BSTR bstrMenuItem,
 [in] BSTR bstrMenuPrompt,
 [out] long* pnMenuID);
HRESULT RemoveTool ([in] long nMenuID);

// UI Password change property
 [propget]
HRESULT EnableUIPasswordChange([out, retval] BOOL *pVal);
[propput]
HRESULT EnableUIPasswordChange([in] BOOL newVal);
};

dispinterface _AppEvents
{
properties:
methods:
 HRESULT OnToolExecute ([in] long nMenuID);
 HRESULT OnUserLogoff();
 HRESULT OnUserExit ();
}

coclass Application
{
[default]_Application;
[source] _AppEvents;
```

Figure 2.1: T_Application Code Example

This page contains Eclipsys proprietary information and is not to be duplicated or disclosed to unauthorized persons.

2.4.1 Application

2.4.1.1 Visual Basic Syntax

```
CVApplication = app.Application
```

Where:

`CVApplication =` An object of type CVApplication; required

`App =` A valid application object; required

RETURN VALUES

Application has two return values.

Table 2.1: Return Values for Application

HRResult	Description
S_OK	The call succeeded and has returned the expected value or values.
E_OUTOFMEMORY	Unable to allocate memory.

REMARKS

This is a standard automation interface property. It returns the application interface, and is a read-only property.

SAMPLE CODE

```
Dim app as New CVApplication
Dim otherapp as CVApplication
otherapp = app.Application
```

Figure 2.2: Application Code Example

SEE ALSO

Parent

2.4.2 Parent

2.4.2.1 Visual Basic Syntax

```
object = app.Parent
```

Where:

`Object =` A valid object; required

`App =` A valid application object; required

This page contains Eclipsys proprietary information and is not to be duplicated or disclosed to unauthorized persons.

RETURN VALUES

Table 2.2: Return Values for Parent

HRResult	Description
S_OK	The call succeeded and has returned the expected value or values.
E_OUTOFMEMORY	Unable to allocate memory.

REMARKS

This is a standard automation interface property. It returns the parent IDispatch interface, and is a read-only property.

SAMPLE CODE

```
Dim app as New CVApplication
Dim parent as Object
parent = app.Parent
```

Figure 2.3: Parent Code Example

SEE ALSO

Application

2.4.3 Name

2.4.3.1 Visual Basic Syntax

```
String = app.Name
```

Where:

String = A string containing the name of the application; required

App = A valid application object; required

RETURN VALUES

Name has two return values.

Table 2.3: Return Values for Name

HRResult	Description
S_OK	The call succeeded and has returned the expected value or values.
E_OUTOFMEMORY	Unable to allocate memory.

REMARKS

This is a standard automation interface property. It returns the name of the application (in this case, Clinical Manager), and is a read-only property.

This page contains Eclipsys proprietary information and is not to be duplicated or disclosed to unauthorized persons.

SAMPLE CODE

```
Dim app as New CVApplication
Dim appName as String
appName = app.Name
```

Figure 2.4: Name Code Example

SEE ALSO

FullName

2.4.4 FullName

2.4.4.1 Visual Basic Syntax

```
String = app.FullName
```

Where:

String = A string containing the full path name of the application; required

App = A valid application object; required

RETURN VALUES

FullName has two return values.

Table 2.4: Return Values for FullName

HRResult	Description
S_OK	The call succeeded and has returned the expected value or values.
E_OUTOFMEMORY	Unable to allocate memory.

REMARKS

This is a standard automation interface property. It returns the full application path name for the application, and is a read-only property.

SAMPLE CODE

```
Dim app as New CVApplication
Dim path as String
path = app.FullName
```

Figure 2.5: FullName Code Example

SEE ALSO

Name

This page contains Eclipsys proprietary information and is not to be duplicated or disclosed to unauthorized persons.

2.4.5 Visible

2.4.5.1 Visual Basic Syntax

```
app.Visible = Boolean
Boolean = app.Visible
```

Where:

App = A valid application object; required

Boolean = A value of Visible property; required

RETURN VALUES

Visible has three return values.

Table 2.5: Return Values for Visible

HRResult	Description
S_OK	The call succeeded and has returned the expected value or values.
E_OUTOFMEMORY	Unable to allocate memory.
E_FAIL	Cannot display application until user logs on.

REMARKS

This is a standard automation interface property. If this property is True, the application appears on the desktop. If False, the application is hidden. The default behavior is hidden.

SAMPLE CODE

```
Dim app as New CVApplication
Dim visible as Boolean
visible = app.Visible
If NOT Visible Then
app.Visible = True
End If
```

Figure 2.6: Visible Code Example

2.4.6 Quit

2.4.6.1 Visual Basic Syntax

```
app.Quit
```

Where:

App = A valid application object; required

This page contains Eclipsys proprietary information and is not to be duplicated or disclosed to unauthorized persons.

RETURN VALUES

Quit has three return values.

Table 2.6: Return Values for Quit

HRResult	Description
S_OK	The call succeeded and has returned the expected value or values.
E_OUTOFMEMORY	Unable to allocate memory.
E_FAIL	Cannot quit because user remains logged on. Can only quit after user has logged off.

REMARKS

This is a standard automation interface method. It does not allow SCM to quit, unless the user has logged off their session.

SAMPLE CODE

```
Dim app as New CVApplication
...
app.Quit
```

Figure 2.7: Quit Code Example

2.4.7 SessionID

2.4.7.1 Visual Basic Syntax

```
app.SessionID = Number
Number = app.SessionID
```

Where:

App = A valid application object; required

Number = Any 16-bit value other than zero

RETURN VALUES

SessionID has two return values.

Table 2.7: Return Values for SessionID

HRResult	Description
S_OK	The call succeeded and has returned the expected value or values.
E_OUTOFMEMORY	Unable to allocate memory.

This page contains Eclipsys proprietary information and is not to be duplicated or disclosed to unauthorized persons.

REMARKS

This is a specialized SCM automation interface property. It provides the session ID for the current context object session. Setting this property makes SCM attempt to connect to the session of the context object identified by the session ID. The session property must be set prior to the first logon attempt with the application object in order to connect to a specific session. After the first logon, the session ID is a read-only property.

SAMPLE CODE

```
Dim app as New CVApplication
If app.SessionID = 0 Then
app.Session = 4
End If
```

Figure 2.8: SessionID Code Example

SEE ALSO

Logon

2.4.8 ActiveTab

2.4.8.1 Visual Basic Syntax

```
Number = app.ActiveTab
app.ActiveTab Number
```

Where:

Number = A valid zero-based tab index; required

App = A valid application object; required

RETURN VALUES

ActiveTab has two return values.

Table 2.8: Return Values for ActiveTab

HRResult	Description
S_OK	The call succeeded and has returned the expected value or values.
E_FAIL	The call indicated a tab number outside the range of the number of tabs.

REMARKS

This is a specialized SCM automation interface property. This property indicates the zero-based index number of the current tab.

This page contains Eclipsys proprietary information and is not to be duplicated or disclosed to unauthorized persons.

SAMPLE CODE

```
Dim app as New CVApplication
Dim tab as CVTab
tab = app.ActiveTab
app.ActiveTab 3
```

Figure 2.9: ActiveTab Code Example

2.4.9 Logon

2.4.9.1 Visual Basic Syntax

```
app.Logon UserName, Password
```

Where:

App = A valid application object; required

UserName = A string containing a SCM username; required

Password = A string containing the user password; required

RETURN VALUES

Logon has two return values.

Table 2.9: Return Values for Logon

HRResult	Description
S_OK	The call succeeded and has returned the expected value or values.
E_OUTOFMEMORY	Unable to allocate memory.

REMARKS

This is a specialized SCM automation interface method. This method must be called before SCM becomes functional for clinical work.

If a session ID has not been set before calling this function, SCM initiates a new session with the context object, and stores the session ID internally.

SAMPLE CODE

```
Dim app as New CVApplication
UserName = “jongmore”
Password = “aaaaaa”
app.Logon jongmore, aaaaaa
```

Figure 2.10: Logon Code Example

SEE ALSO

Logoff, SessionID

This page contains Eclipsys proprietary information and is not to be duplicated or disclosed to unauthorized persons.

2.4.10 Logoff

2.4.10.1 Visual Basic Syntax

```
app.Logoff
```

Where:

App = A valid application object; required

RETURN VALUES

Logoff has two return values.

Table 2.10: Return Values for Logoff

HRResult	Description
S_OK	The call succeeded and has returned the expected value or values.
E_FAIL	User has unsaved work outstanding in the SCM session.

REMARKS

This is a specialized SCM automation interface method. This method logs a user off a running session of SCM.

The functionality is identical to choosing the Logoff option in the user interface; however, the Logon window does not appear while SCM is running in automation controller mode.

SAMPLE CODE

```
Dim app as New CVApplication
UserName = “jongmore”
Password = “aaaaaa”
app.Logon jongmore, aaaaaa
...
app.Logoff
```

Figure 2.11: Logoff Code Example

SEE ALSO

Logon

This page contains Eclipsys proprietary information and is not to be duplicated or disclosed to unauthorized persons.

2.4.11 ChangePassword

2.4.11.1 Visual Basic Syntax

```
app.ChangePassword OldPassword, NewPassword
```

Where:

App = A valid application object; required

OldPassword = The old user password; required

NewPassword = The new user password; required

RETURN VALUES

ChangePassword has two return values.

Table 2.11: Return Values for ChangePassword

HRResult	Description
S_OK	The call succeeded and has returned the expected value or values.
E_FAIL	There is no currently logged on user or invalid password parameters.

REMARKS

This is a specialized SCM automation interface method. This method must be called during a valid SCM session (in other words, a user must be logged on). This method changes the password for the currently logged-on user.

SAMPLE CODE

```
Dim app as New CVApplication
UserName = “jongmore”
Password = “aaaaaa”
app.Logon jongmore, aaaaaa
app.ChangePassword aaaaaa, bbbbbb
```

Figure 2.12: ChangePassword Code Example

2.4.12 AddTrigger

2.4.12.1 Visual Basic Syntax

```
Number = app.AddTrigger (eventsink)
```

Where:

Number = The number representing the unique ID for the dispatch interface; required

App = A valid application object; required

Eventsink = A valid COM object implementing the AppEvent interface; required

This page contains Eclipsys proprietary information and is not to be duplicated or disclosed to unauthorized persons.

Return Values

AddTrigger has four return values.

Table 2.12: Return Values for AddTrigger

HRResult	Description
S_OK	The call succeeded and has returned the expected value or values.
E_FAIL	There is no currently logged on user or there are invalid password parameters.
E_INVALIDARG	Eventsink is an invalid object.
CONNECT_E_ADVISELIMIT	Exceeded the limit of allowable triggers.

Remarks

This is a specialized SCM automation interface method. This method provides the ability to use a COM object implementing the AppEvent interface in automation programming languages that do not support connection points or the WithEvents keyword (for example, VBA, VBScript).

This method can be implemented and used in addition to connection points. Only one AppEvent interface is allowed per application object.

Sample Code

```
Dim app as New CareVISION.Application
Dim event as Healthcare.AppEvent
Dim cookie as Number
...
cookie = app.AddTrigger(event)
...
Dim menuID as Number
menuID = app.AddTool(“&DMS”, “Go to Disease Management System”)
...
app.RemoveTool menuID
app.RemoveTrigger cookie
```

Figure 2.13: AddTrigger Code Example

See also

RemoveTrigger, AddTool, RemoveTool

This page contains Eclipsys proprietary information and is not to be duplicated or disclosed to unauthorized persons.

2.4.13 RemoveTrigger

2.4.13.1 Visual Basic Syntax

```
app.RemoveTrigger Number
```

Where:

App = A valid application object; required

Number = The number representing the unique ID for the dispatch interface; required

RETURN VALUES

RemoveTrigger has three return values.

Table 2.13: Return Values for RemoveTrigger

HRResult	Description
S_OK	The call succeeded and has returned the expected value or values.
E_INVALIDARG	Eventsink is an invalid object.
CONNECT_E_NOCONNECTION	No existing triggers attached, or invalid ID for dispatch interface.

REMARKS

This is a specialized SCM automation interface method. This method removes an AppEvent interface from the application object. This method can remove a previously installed trigger only using its unique ID.

SAMPLE CODE

```
Dim app as New CareVISION.Application
Dim event as Healthcare.AppEvent
Dim cookie as Number
...
cookie = app.AddTrigger(event)
...
Dim menuID as Number
menuID = app.AddTool("&DMS", "Go to Disease Management System")
...
app.RemoveTool menuID
app.RemoveTrigger cookie
```

Figure 2.14: RemoveTrigger Code Example

SEE ALSO

AddTrigger, AddTool, RemoveTool

This page contains Eclipsys proprietary information and is not to be duplicated or disclosed to unauthorized persons.

2.4.14 AddTool

2.4.14.1 Visual Basic Syntax

```
Number = app.AddTool (MenuItem, MenuPrompt)
```

Where:

Number =	The number representing the menu item's unique ID; required
App =	A valid application object; required
MenuItem =	The string containing the menu item name; required
MenuPrompt =	The string containing the status bar message for the menu item; required

RETURN VALUES

AddTool has two return values.

Table 2.14: Retrun Values for AddTool

HRResult	Description
S_OK	The call succeeded and returns the expected value or values.
E_FAIL	Exceeded the limit of tool entries for the menu.

REMARKS

This is a specialized SCM automation interface method. A connection point must be made with this interface before calling this method. The menu ID returned is the actual menu ID used by SCM.

This menu ID is passed as a parameter to the supplied connection point to indicate which menu item was chosen.

SAMPLE CODE

```
Dim WithEvents app as CareVISION.Application
Set app = New CareVISION.Application
Dim menuID as Number
menuID = app.AddTool("&DMS", "Go to Disease Management System")
...
app.RemoveTool(menuID)
```

Figure 2.15: AddTool Code Example

SEE ALSO

RemoveTool

This page contains Eclipsys proprietary information and is not to be duplicated or disclosed to unauthorized persons.

2.4.15 RemoveTool

2.4.15.1 Visual Basic Syntax

```
app.RemoveTool MenuID
```

Where:

App = A valid application object; required

MenuId = The menuID returned from the AddTool method; required

RETURN VALUES

RemoveTool has two return values.

Table 2.15: Return Values for RemoveTool

HRResult	Description
S_OK	The call succeeded and returns the expected value or values.
E_FAIL	MenuID is invalid. There is no registered tool for that ID.

REMARKS

This is a specialized SCM automation interface method. Any value returned by an AddTool method call is a valid argument. Can only be called after tools have been added.

SAMPLE CODE

```
Dim app as New CVApplication
Dim menuID as Number
menuID = app.AddTool("&DMS", "Go to Disease Management System")
...
app.RemoveTool(menuID)
```

Figure 2.16: RemoveTool Code Example

SEE ALSO

AddTool

2.4.16 EnableUIPasswordChange

2.4.16.1 Visual Basic Syntax

```
app.EnableUIPasswordChange = True
boolean = app.EnableUIPasswordChange
```

Where:

App = A valid application object; required

Boolean = Boolean variable for setting or getting the value; required

This page contains Eclipsys proprietary information and is not to be duplicated or disclosed to unauthorized persons.

RETURN VALUES

EnableUIPasswordChange has two return values.

Table 2.16: Return Values for EnableUIPasswordChange

HRResult	Description
S_OK	The call succeeded and has returned the expected value or values.
E_OUTOFMEMORY	Unable to allocate memory.

REMARKS

This is a specialized SCM automation interface property. This property determines whether the Change Password menu option is enabled or disabled. This check is in addition to any SCM rights checking.

For the menu item to appear, the user must have the right to change the password and have EnableUIPasswordChange property set to True. The default value for this property is False.

SAMPLE CODE

```
Dim app as New CVApplication
Dim CanChangePassword as Boolean
CanChangePassword = app.EnableUIPasswordChange
If (!CanChangePassword) Then
app.EnableUIPasswordChange = True
End If
```

Figure 2.17: EnableUIPasswordChange Code Example

2.4.17 Application_OnToolExecute(nMenuID)

REMARKS

This event is called when SCM selects one of the tool methods added by a call to AddTool. The nMenuID item is used to determine which tool item was chosen.

SAMPLE CODE

```
Dim app as New CVApplication
...
Sub app_OnToolExecute(MenuID)
Select Case MenuID
Case DMSID
 LaunchDMS
Case PACSID
 LaunchPACS
Case Else
 MsgBox ("Tool not available")
End Select
End Sub
```

Figure 2.18: Application_OnToolExecute Code Example

This page contains Eclipsys proprietary information and is not to be duplicated or disclosed to unauthorized persons.

SEE ALSO

AddTool, RemoveTool

2.4.18 Application_OnUserExit( )

REMARKS

This event is called when the SCM user attempts to exit a SCM instance launched by an automation controller. This enables the controller to determine how to respond to the user's action.

SAMPLE CODE

```
Dim app as New CVApplication
...
Sub app_OnUserExit
' Logoff user
app.Logoff
'Present this application
this.Visible = True
End Select
End Sub
```

Figure 2.19: Application_OnUserExit Code Example

SEE ALSO

Application_OnUserLogoff( )

2.4.19 Application_OnUserLogoff( )

REMARKS

This event is called when the SCM user attempts to log off a SCM instance launched by an automation controller. This enables the controller to determine how to respond to the user's action.

SAMPLE CODE

```
Dim app as New CVApplication
...
Sub app_OnUserLogoff
' Logoff user
app.Logoff
'Present this application
this.Visible = True
End Select
End Sub
```

Figure 2.20: Application_OnUserLogoff Code Example

SEE ALSO

Application_OnUserExit( )

This page contains Eclipsys proprietary information and is not to be duplicated or disclosed to unauthorized persons.

2.5 HL7 Desktop Integration Support

SCM has the ability to exchange clinical context data with other healthcare applications on the desktop. Starting with SCM, this can be done in one of two ways. First, the existing proprietary HVCContextServer method is currently supported (but may not be supported in future releases). SCM can now adhere to the HL7 CCOW standard for desktop integration. This is the preferred method, and can be accessed using the HL7 desktop interface, which is described in Desktop Integration in Sunrise Clinical Manager.

2.5.1 IHL7Desktop Interface

This is the new interface used for HL7 desktop integration. It derives from the default _Application interface so that all functionality exposed by _Application can also be accessed through IHL7Desktop.

```
interface IHL7Desktop : _Application
{
// The HL7DesktopState property determines the type of
// context participant SCM plays on the HL7 Desktop
[propget, id(18), helpstring("property HL7DesktopState")]
HRESULT HL7DesktopState([out, retval] scmHL7DesktopState
*pState);
[propput, id(18), helpstring("property HL7DesktopState")]
HRESULT HL7DesktopState([in] scmHL7DesktopState newState);

// Allows SCM to join or leave the HL7 Desktop as a
// context participant
[id(19), helpstring("method JoinContext")]
HRESULT JoinHL7Desktop();
[id(20), helpstring("method LeaveContext")]
HRESULT LeaveHL7Desktop();

// The AllowDataLossOnContextChange property determines
whether
// unsubmitted data is allowed to be discarded in response
// to a context change, including logout.
[propget, id(21), helpstring("property
AllowDataLossOnContextChange")]
HRESULT AllowDataLossOnContextChange([out, retval] BOOL
*pbAllow);
[propput, id(21), helpstring("propertY
AllowDataLossOnContextChange")]
HRESULT AllowDataLossOnContextChange([in] BOOL bAllow);
};
```

Figure 2.21: IHL7Desktop Interface Code Example

This page contains Eclipsys proprietary information and is not to be duplicated or disclosed to unauthorized persons.

2.5.2 HL7DesktopState

2.5.2.1 Visual Basic Syntax

```
desktop.HL7DesktopState = State
State = desktop.HL7DesktopState
```

Where:

Desktop = A valid reference to the IHL7Desktop interface; required

State = A variable of enumerated type scmHL7DesktopState; required

The valid values include `HL7DesktopState_None`, `HL7DesktopState_Peer`, and `HL7DesktopState_Master`.

RETURN VALUES

HL7DesktopState has two return values.

Table 2.17: Return Values for HL7DesktopState

HRResult	Description
S_OK	The call succeeded and has returned the expected value or values.
E_INVALIDARG	The call attempted to set the state to an invalid value.

REMARKS

This is a standard automation interface property. This property defaults to HL7Desktop_None, which means HL7 desktop integration is not being used.

It is necessary to set the property to the appropriate value prior to calling JoinHL7Desktop().

This page contains Eclipsys proprietary information and is not to be duplicated or disclosed to unauthorized persons.

Table 2.18: HL7DesktopState Table

Setting HL7DesktopState =	Means
HL7DesktopState_None	Default value. No HL7 desktop integration.
HL7DesktopState_Peer	SCM publishes and responds to user, patient, and view subject changes. Only one instance of SCM can be run in this mode. If another SCM application is started, the prior SCM instance is suspended. If another participating application does not use patient visit IDs, SCM responds to patient context changes by searching on the patient MRN only. However, SCM always publishes both the MRN and visit ID.
HL7DesktopState_Master	SCM runs as master participant. It publishes user, patient, and view context data. To allow certain slave applications to set control back to SCM, it responds to view context changes. It does not respond to user and patient changes in this mode.

Table 2.19 shows what SCM publishes when running as a certain type of participant:

Table 2.19: Participant Table

	User	Patient	View
peer	X	X	X
master	X	X	X

Table 2.20 shows what SCM responds to when running as a certain type of participant:

Table 2.20: Participant Table

	User	Patient	View
peer	X	X	X
master			X

SAMPLE CODE

```
Dim WithEvents scm As CareVISION.Application
Dim m_enumDesktopState As CareVISION.scmHL7DesktopState
Dim desktop As CareVISION.IHL7Desktop

' Create a new instance of scm
Set scm = New CareVISION.Application

' Initialize the desktop state
```


This page contains Eclipsys proprietary information and is not to be duplicated or disclosed to unauthorized persons.

```
Set desktop = scm
desktop.HL7DesktopState = HL7DesktopState_Master
```

Figure 2.22: HL7DesktopState Code Example

2.5.3 JoinHL7Desktop

2.5.3.1 Visual Basic Syntax

```
desktop.JoinHL7Desktop
```

Where:

Desktop = A valid reference to the IHL7Desktop interface; required

RETURN VALUES

JoinHL7Desktop has two return values.

Table 2.21: Return Values for JoinHL7Desktop

HRResult	Description
S_OK	The call succeeded and has returned the expected value or values.
E_FAILED	Unable to join the desktop.

REMARKS

This is a specialized SCM automation interface method. This method must be called to establish the link to the desktop prior to logging on to SCM.

SAMPLE CODE

```
Dim WithEvents scm As CareVISION.Application
Dim m_enumDesktopState As CareVISION.scmHL7DesktopState
Dim desktop As CareVISION.IHL7Desktop

' Create a new instance of scm
Set scm = New CareVISION.Application

' Initialize the desktop state
Set desktop = scm

' join the HL7 desktop
desktop.JoinHL7Desktop
```

Figure 2.23: JoinHL7Desktop Code Example

SEE ALSO

LeaveHL7Desktop

This page contains Eclipsys proprietary information and is not to be duplicated or disclosed to unauthorized persons.

2.5.4 LeaveHL7Desktop

2.5.4.1 Visual Basic Syntax

```
desktop.LeaveHL7Desktop
```

Where:

desktop = A valid reference to the IHL7Desktop interface; required

RETURN VALUES

LeaveHL7Desktop has two return values.

Table 2.22: Return Values for LeaveHL7Desktop

HRResult	Description
S_OK	The call succeeded and has returned the expected value or values.
E_FAILED	Unable to leave the desktop.

REMARKS

This is a specialized SCM automation interface method. This method must be called after logging off SCM. It breaks the clinical link to the desktop, and performs any necessary cleanup.

SAMPLE CODE

```
Dim WithEvents scm As CareVISION.Application
Dim m_enumDesktopState As CareVISION.scmHL7DesktopState
Dim desktop As CareVISION.IHL7Desktop

' Create a new instance of scm
Set scm = New CareVISION.Application

' Initialize the desktop state
Set desktop = scm

' join the HL7 desktop
desktop.LeaveHL7Desktop
```

Figure 2.24: LeaveHL7Desktop Code Example

SEE ALSO

JoinHL7Desktop

This page contains Eclipsys proprietary information and is not to be duplicated or disclosed to unauthorized persons.

2.5.5 AllowDataLossOnContextChange

2.5.5.1 Visual Basic Syntax

```
desktop.AllowDataLossOnContextChange = bAllow
bAllow = desktop.AllowDataLossOnContextChange
```

Where:

Desktop = A valid reference to the IHL7Desktop interface; required

bAllow = A Boolean variable

RETURN VALUES

AllowDataLossOnContextChange has two return values.

Table 2.23: Return Values for AllowDataLossOnContextChange

HRResult	Description
S_OK	The call succeeded and has returned the expected value or values.
E_OUTOFMEMORY	Unable to allocate memory.

REMARKS

The default value is False, which means SCM does not discard unsubmitted data as a result of a context change or at logoff.

SAMPLE CODE

```
Dim WithEvents scm As CareVISION.Application
Dim m_enumDesktopState As CareVISION.scmHL7DesktopState
Dim desktop As CareVISION.IHL7Desktop

' Create a new instance of scm
Set scm = New CareVISION.Application

' Initialize the data loss property
Set desktop = scm
desktop.AllowDataLossOnContextChange = True
```

Figure 2.25: AllowDataLossOnContextChange Code Example

This page contains Eclipsys proprietary information and is not to be duplicated or disclosed to unauthorized persons.

CHAPTER 3: CONTEXT SERVER

Contents

3.1 Overview 31
3.2 Data Representation 32
3.3 Context Server Installation and Configuration 33
3.4 Referencing the Classes and Interfaces 35
3.5 The Context Server Interfaces 36
3.6 Objects 57

3.1 Overview

This chapter describes the Sunrise Clinical Manager context server (HVCContextServer.exe) implementation and provides information to developers of healthcare applications on how to subscribe to context information from SCM, and to publish context information to SCM. While this method is currently supported, see the method described in "Desktop Integration in Sunrise Clinical Manager," which is the preferred method.

The context server allows SCM to share context information with other Windows desktop applications. Context information is information based on the current operating state of all co-operating applications. Sharing context information enhances the interoperability between SCM and other applications.

The technology for sharing context information is the Component Object Model (COM), the standard for advanced object-based inter-process communications on Windows platforms. By defining COM objects containing necessary context data, it is possible for any number of applications to co-operate on a single desktop.

The context server is composed of three COM objects:

- The Context Manager: Manages the changes to the context data, and notifies context participants of changes in context data.
- The Context Data: Contains the data that represents the current operating status of all shared applications.
- The Context Participant: An event sink implemented by applications interested in reviewing notification messages from the context manager when the context data changes.

This page contains Eclipsys proprietary information and is not to be duplicated or disclosed to unauthorized persons.

The context manager and context data objects are implemented in a stand alone executable that runs on the SCM client. The context participant interface object is implemented in SCM and by other healthcare products interested in sharing operating context. All interfaces return a common set of HRESULTs. The IErrorInfo interface is supported for obtaining extended error information.

3.2 Data Representation

There are two types of data stored by the context object:

- Participant state information: This includes the participant token, which session the user is joined to, the participant process ID, and various status indicators, such as InTransaction.
- Session data: Each session has two lists of context data: a current context and a next context.

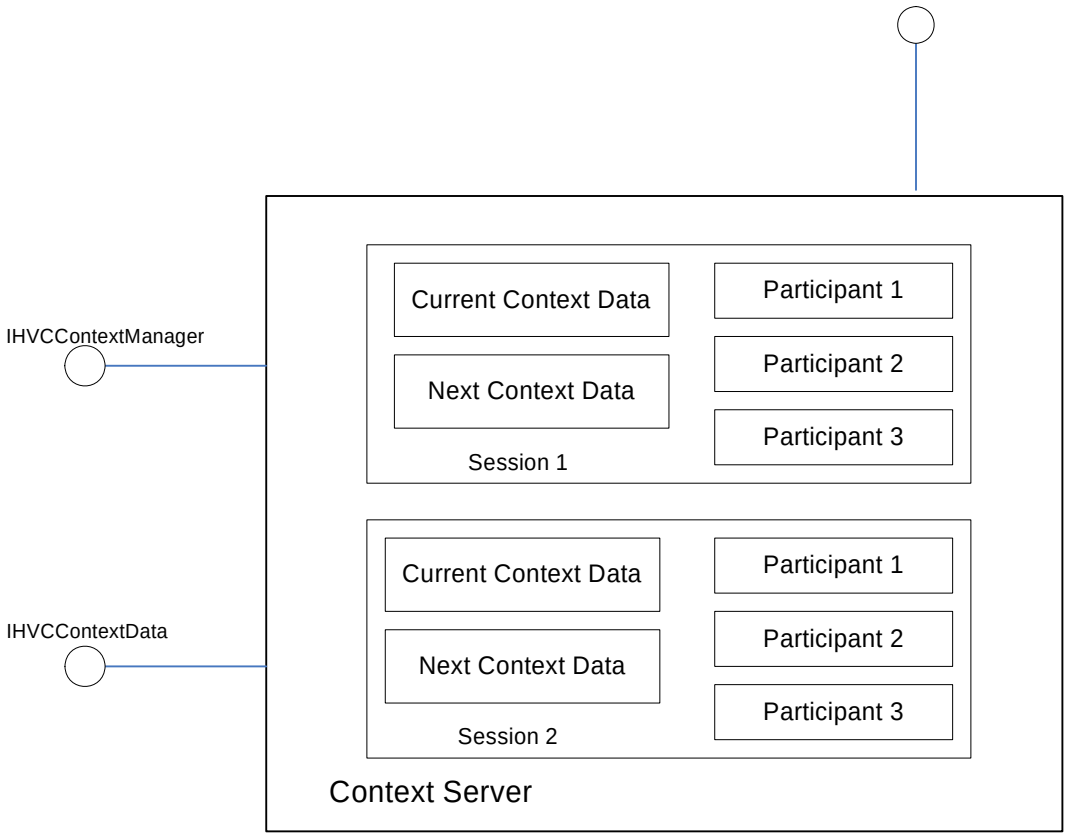


Figure 3.1: Data Representation

The context server is implemented using COM. Clients use the IHVCContextManager and IHVCContextData interfaces to communicate with the server. IHVCContextManager and IHVCContextData are implemented on the

This page contains Eclipsys proprietary information and is not to be duplicated or disclosed to unauthorized persons.

same object. In Sunrise 4.0 XA, the IHVCContextManager2 interface can be used instead of IHVCContextManager.

The server is implemented with a fixed array of sessions. Each session has an array of two context data objects, and a fixed array of participant objects.

3.2.1 Session Control

A session is created at the request of a client using the IHVCContextManager interface method CreateContextSession. A session is created with no participants. Each participant must join the session using the JoinContextSession method. The session maintains the participants joined to it.

Each session is able to recover from participant failure. There are several ways a participant can fail:

- The application shuts down without notifying the server.
- The application freezes (infinite loop or other crash that does not remove the application from the active process list). If this occurs during a notification, the participant is marked as having an outstanding notification. In this case, the notification is retried until the participant shuts down, is successfully notified, or a new context change is published.
- The application is too slow (starts a transaction but does not end it in a timely manner). Each session has a timer associated with it that automatically closes an open transaction. This causes the caller to get an error when they attempt to call PublishContextChanges.
- The participant sends invalid data to server, such as an invalid token or badly formed context data.

3.3 Context Server Installation and Configuration

The HVCContextServer application (HVCContextServer.exe) is installed on each SCM workstation during the client installation. No special installation is required.

The context server is a self-registering executable, and should be automatically registered by the SCM Setup program. If registering or unregistering it on a workstation, use the following steps.

Note: HVCContextServer must be run from the directory where it resides, if the location is not in the path. The default location is C:\Program Files\Common Files\HealthVISION.

◆ **To register the context server:**

▸ Run the program using:

```
HVCContextServer/REGSERVER
```


This page contains Eclipsys proprietary information and is not to be duplicated or disclosed to unauthorized persons.

◆ To unregister the context server:

▶ Run the program using:

```
HVCContextServer/UNREGSERVER
```

3.3.1 Enabling the Context Server

The context object server allows SCM to integrate with third-party programs. To enable the context server, allowing it to make the connection to the context object, configure the environment profile using the Configuration Tools application. This is an entry that must be created; it is not already present in the list of entries under the Facility section. After creating the entry, set the value to Y if you want SCM to connect with the context object. This must be done for all facilities in the enterprise. For more information on setting environment profile variables, see the *Sunrise Clinical Manager General Configuration Guide*.

◆ To enable the context server:

1 Double-click Dictionary in the Configuration Tools window. The Dictionary Maintenance window opens.

2 In the Categories list, choose Miscellaneous.

3 In the Dictionaries list, double-click Environment Profile. The Environment Profile Maintenance window opens.

4 Select the Facility section.

5 Select Add Sub Level. The Add Environment Profile window opens.

6 From the Profile Type list, select Entry.

7 In the Code field, enter UsingContextServer.

8 In the Description field, enter a phrase that describes the entry's purpose.

9 In the Value field, enter Y.

10 Click OK. Return to the Environment Profile Maintenance window.

11 Click Close.

3.3.2 Starting SCM from the Command Line

To start SCM from the command line, allowing it to join the current session, use these command line parameters:

```
/uusername
/ppassword
/ssessionID
```

For example, to start SCM for the user jongmore with a password of xxxjag:

```
hvc/ujongmore/pxxxjag
```


This page contains Eclipsys proprietary information and is not to be duplicated or disclosed to unauthorized persons.

When SCM is started from the command line, SCM exits when the user changes.

3.4 Referencing the Classes and Interfaces

The class and interface identifiers are embedded in the type library information. The HVCContextServer.exe file contains the type library.

◆ To reference the classes and interfaces:

- From Visual Basic, select the Project/References menu to bring up the References window. Scroll down and select the reference named HVCContextServer 1.0 Type Library. If it is not listed, click Browse and locate the HVCContextServer.exe file. It is typically located in the C:\Program Files\Common Files\HealthVISION folder. After it is added, use the Object Browser tool to examine the methods and parameters. In the Project/Library field, change <All Libraries> to HVCCONTEXTSERVERLib.
- From Visual C++ use the #import directive to create the two files HVCContextServer.tlh and HVCContextServer.tli. These files implement a class wrapper for all the classes and interfaces used by the context server. For more information on the #import directive, see the Visual C++ online help.

To directly access the interfaces, get the required information from the type library using the OLE/COM Object Viewer application that ships with Microsoft Developer Studio.

```
Global theContext As New HVCCONTEXTSERVERLib.CoContextManager
Global thecontextData As HVCCONTEXTSERVERLib.IHVCContextData
Global theSessionToken As Long

Public Sub Main()

theSessionToken = theContest.CreateContestSession

' Query the CoContextManager object for the IHVCContextData
interface.
Set theContextData = theContext

' other installation

End Sub
Using Visual C++:
// example.cpp : implementation file
//
#include "stdafx.h"
#include "example.h"

#import "..\hvccontextserver.exe" no_namespace

MainFunction()
{
CoInitialize(NULL);
```


This page contains Eclipsys proprietary information and is not to be duplicated or disclosed to unauthorized persons.

```
IHVCContextManagerPtrpContextManager;
// Smart pointer created by #import
HRESULThr;

hr = CoCreateInstance( __uuidof( CoContextManager ), // guids
defined by #import
 NULL,
 CLSCTX_LOCAL_SERVER,
 __uuidof( IHVCContextManager ),
  (void**)&pContextManager );

long  nSessionToken;

// Call the create session interface, this wrapper method was
// created by the #import directive
// The wrapper methods throw an _com_error object if the call
// fails, this object contains the HRESULT, the interface
// pointer and the interface GUID.

try
{
 nSessionToken = pContextManager->CreateContextSession();
}
 catch( _com_error& theError )
{
 // Handle errors
}

// Additional stuff

CoUninitialize();
}
```

Figure 3.2: Visual Basic Code Example

3.5 The Context Server Interfaces

There are four context server interfaces:

- IHVCContextManager.
- IHVCContextManager2.
- IHVCContextData.
- IHVCContextParticipant.

3.5.1 IHVCContextManager

This interface manages the sessions and data transactions within the context server.

Figure 3.3: IHVCContextManager Interface

This page contains Eclipsys proprietary information and is not to be duplicated or disclosed to unauthorized persons.

The IHVCContextManager2 interface extends the IHVCContextManager interface by adding the GetApplicationSessionsAndPIDs() method. This is similar to the existing GetApplicationSessions() method except it also returns the process ID of each of the applications.

Note: For backward-compatibility, IHVCContextManager is still the default interface.

```
interface IHVCContextManager: IDispatch
{
HRESULT GetCurrentContextToken([in] long nSessionToken,
 [out,retval] long* pnContextToken);
HRESULT CreateContextSession([out,retval] long*
pnSessionToken);
HRESULT JoinContextSession([in] long nSessionToken,
 [in] long nProcessID,
 [in] VARIANT_BOOL bWaitForTransaction,
 [in] LPDISPATCH pdispParticipant,
 [out,retval] long* pnParticipant);
HRESULT LeaveContextSession([in] long nParticipant);
HRESULT GetApplicationSessions([out] LPVARIANT pvtAppNames,
 [out] LPVARIANT pvtAppSessions);
HRESULT BeginContextChanges([in] long nParticipant,
 [out, retval] long* pnContext);
HRESULT PublishContextChanges([in] long nContext,
 [in] VARIANT_BOOL bAccept);
HRESULT TakeOwnership([in] long nParticipant);
HRESULT ReleaseOwnership([in] long nParticipantToken);
HRESULT FreezeContextSession([in] long nProcessID, [in] long
nSession);
HRESULT ReleaseSession([in] long nProcessID, [in] long
nSession);
};
interface IHVCContextManager2 : IHVCContextManager
{
 [id(12), helpstring("method GetApplicationSessionsAndPIDs")]
    HRESULT GetApplicationSessionsAndPIDs([out] VARIANT*
pvarApplicationNames, [out] VARIANT* pvarApplicationSessions,
[out] VARIANT* pvarParticipantPIDs);
};
```

Figure 3.4: IHVCContextManager2 Interface

This page contains Eclipsys proprietary information and is not to be duplicated or disclosed to unauthorized persons.

3.5.2 GetCurrentContextToken

3.5.2.1 Visual Basic Syntax

```
nContextToken=
ContextManager.GetCurrentContextToken(nSessionToken)
```

Where:

nContextToken = The identifier for the current context used in IcontextData methods

nSessionToken = A valid session ID from an existing context session; required

ContextManager = Reference to IHVCContextManager interface; required

RETURN VALUES

GetCurrentContextToken has two return values.

Table 3.1: Return Values for GetCurrentContextToken

HRResult	Description
S_OK	The call succeeded and has returned the expected value or values.
0x80000302 (2147484418) : Invalid session token	Session token does not identify an existing context session.

REMARKS

An application can get a context token from an existing session token with this function. In this way, an application can then get access to the context manager data without joining the context session.

SAMPLE CODE

```
Dim CMObject as New CoContextManager
Dim ifContextManager as IHVCContextManager
Dim ifContextData as IHVCContextData
Set ifContextManager = CoContextManager
Set ifContextData = CoContextManager
...
GetApplicationSessions(vtAppNames, vtAppSessions)
...
nContextToken =
ifContextManager.GetCurrentContextToken(vtAppSessions[1])
...
vtContextItems = ifContextData.GetContextItems(nContextToken)
...
vtContextValues = ifContextData.GetContextValues(nContextToken, _
vtContextItems)
See also

IHVCContextData.GetContextItems, IHVCContextData.GetContextValues
```

Figure 3.5: GetCurrentContextToken Code Example

This page contains Eclipsys proprietary information and is not to be duplicated or disclosed to unauthorized persons.

3.5.3 CreateContextSession

3.5.3.1 Visual Basic Syntax

```
nSession =ContextManager.CreateContextSession
```

Where:

nSession = A long variable containing the session identifier; required

ContextManager = A valid context manager interface; required

RETURN VALUES

CreateContextSession has two return values.

Table 3.2: Return Values for CreateContextSession

HRResult	Description
S_OK	The call succeeded and has returned the expected value or values.
0x80000300 (2147484416) : Too many sessions	Maximum number of sessions has been reached.

REMARKS

This method creates a new instance of a context session. The new context session is automatically cleaned up if no participant joins the session within a 10-second time period.

SAMPLE CODE

```
Dim CMObject as New CoContextManager
Dim ifContextManager as IHVCContextManager
Set ifContextManager = CoContextManager
...
Dim nSession As Long
nSession = ifContextManager.CreateContextSession()
```

Figure 3.6: CreateContextSession Code Example

SEE ALSO

JoinContextSession, LeaveContextSession

3.5.4 JoinContextSession

3.5.4.1 Visual Basic Syntax

```
nParticipant= ContextManager.JoinContextSession (nSessionToken,
nProcessID, bWaitForTransaction, itfContextParticipant)
```


This page contains Eclipsys proprietary information and is not to be duplicated or disclosed to unauthorized persons.

Where:

nParticipant =	A long value containing the participant's identifier; required
ContextManager =	A valid context manager interface; required
nSessionToken =	A long value containing a valid session identifier; required
nProcessID =	A long value containing the process ID of the participating application; required
bWaitForTransaction =	A Boolean value indicating whether the application is willing to wait for a context change to complete before joining the session; required
itfContextParticipant =	An Idispatch-based interface representing the participant object. This interface is used for participant callback; required

Return Values

JoinContextSession has four return values.

Table 3.3: Return Values for JoinContextSession

HRResult	Description
S_OK	The call succeeded and has returned the expected value or values.
0x80000302 (2147484418) : Invalid Session Token	Session token does not identify an existing context session.
0x80000313 (2147484435): Invalid Process ID	Process ID not found in task list.
0x80000303 (2147484419): Too many participants	Session token does not identify an existing context session.

Remarks

This method adds a participant to an existing context session. Each participant has a unique identifier, as the session token is encoded as part of the participant token. The participant callback object is passed in and added to the list of participants to be informed of data changes in that session. The participant's process ID is passed in to allow the context manager to query the application for existence, and obtain the application's main window title. The method GetApplicationSessions returns the window titles of all participating applications. There is a 10-second window to complete data transactions following a call to CreateContextSession. If the timer closes a new session, the error Invalid Session Token is returned.

Sample Code

```
Dim CMObject as New CoContextManager
Dim ifContextManager as IHVCContextManager
Dim ifParticipant as New CoParticipant
```


This page contains Eclipsys proprietary information and is not to be duplicated or disclosed to unauthorized persons.

```
Set ifContextManager = CoContextManager
...
Dim nSession As Long
nSession = 65536 'This value comes from existing session
nParticipant = ifContextManager.JoinContextSession(nSession,
GetProcessID(), _
True, ifParticipant)
```

Figure 3.7: JoinContextSession Code Example

PARTICIPANT CALLBACK OBJECT

This is a new class module that must be added. In the sample code above, it is named Co-Participant and is responsible for receiving notifications of context data changes. Set the Instancing property of this class to 2 - PublicNotCreatable.

Add this code to the class module:

```
Implements CoContextParticipant
Private Sub CoContextParticipant_ContextChanged(ByVal
nContextToken As Long)
' The context data has changed.  Retrieve the latest data using
nContextToken and perform any other tasks specific to the
application.
End Sub
```

Figure 3.8: Participant Callback Object

SEE ALSO

CreateContextSession, LeaveContextSession

3.5.5 LeaveContextSession

3.5.5.1 Visual Basic Syntax

```
ContextManager.LeaveContextSession nParticipant
```

Where:

ContextManager = A valid context manager interface; required

nParticipant = A long value identifying a valid participant token; required

RETURN VALUES

LeaveContextSession has two return values.

Table 3.4: Return Values for LeaveContextSession

HRResult	Description
S_OK	The call succeeded and has returned the expected value or values.
0x80000302 (2147484418) : Invalid participant token	Participant token is not valid.

This page contains Eclipsys proprietary information and is not to be duplicated or disclosed to unauthorized persons.

REMARKS

This method removes a participant from the context change notification system. The server cleans up any data owned by the participant. This method maintains a cross-reference of the participant, session, and process ID.

SAMPLE CODE

```
Dim CMObject as New CoContextManager
Dim ifContextManager as IHVCContextManager
Dim ifParticipant as New CoParticipant
Set ifContextManager = CoContextManager
...
Dim nSession As Long
nSession = 65536 'This value comes from existing session
nParticipant = ifContextManager.JoinContextSession(nSession,
GetProcessID(), _
True, ifParticipant)
...
ifContextManager.LeaveContextSession nParticipant
```

Figure 3.9: LeaveContextSession Code Example

SEE ALSO

CreateContextSession, JoinContextSession

3.5.6 GetApplicationSessions

3.5.6.1 Visual Basic Syntax

```
ContextManager.GetApplicationSessions(vtAppNames, vtAppSessions)
```

Where:

ContextManager =	A valid context manager interface; required
vtAppNames =	A variant that is populated with an array of application names; required
vtSessionNames =	A variant that is populated with an array of corresponding session IDs; required

RETURN VALUES

GetApplicationSessions has one return value.

Table 3.5: Return Values for GetApplicationSessions

HRResult	Description
S_OK	The call succeeded and has returned the expected value or values.

This page contains Eclipsys proprietary information and is not to be duplicated or disclosed to unauthorized persons.

REMARKS

This method returns a cross-reference list of application main window titles and their associated session IDs. The session IDs are in a format that allows them to be used as session identifiers in the GetCurrentContextToken and JoinContextSession methods.

This method is designed to be used by an application that does not support a command line method of obtaining the Session Token; instead the application presents the user with a list of other applications grouped by their sessions. The user can choose which group to use.

SAMPLE CODE

```
Dim CMObject As New CoContextManager
Dim ifContextManager As IHVCContextManager
Set ifContextManager = CoContextManager
...
Dim vtAppName, vtAppNames As Variant
Dim vtSession, vtSessions As Variant
ifContextManager.GetApplicationSessions vtAppNames, vtSessions
For Each vtAppName In vtAppNames
Debug.Print vtAppName
Next vtAppName
For Each vtSession In vtSessions
Debug.Print vtSession
Next vtAppName
```

Figure 3.10: GetApplicationSessions Code Example

SEE ALSO

JoinContextSession, GetCurrentContextToken, GetApplicationSessionsAndPIDs

3.5.7 GetApplicationSessionsAndPIDs

This method is similar to GetApplicationSessions(). It returns a cross-reference list of application main window titles and their associated session IDs and process IDs.

SAMPLE CODE

```
Private Sub btnRefresh_Click()
    ' Retrieve the list of available sessions
    Dim AppName, AppNames As Variant
    Dim Session, Sessions As Variant
    Dim PartID, PartIDs As Variant
    Dim nIndex As Integer
    Dim SessionEntry As String
    Dim PartIDList As String

On Error GoTo myHandler

    Dim CtxMgr2 As IHVCContextManager2
    Set CtxMgr2 = m_CtxMgr
' Need to use the enhanced interface
If Not CtxMgr2 Is Nothing Then
CtxMgr2.GetApplicationSessionsAndPIDs AppNames, Sessions, PartIDs
lstSessions.Clear
```


This page contains Eclipsys proprietary information and is not to be duplicated or disclosed to unauthorized persons.

```
            nIndex = 0
            For Each AppName In AppNames
                Session = Sessions(nIndex)
                PartID = PartIDs(nIndex)
                SessionEntry = "[" & Session & "] <" & PartID & ">" &
AppName
                lstSessions.AddItem SessionEntry, nIndex
                lstSessions.ItemData(nIndex) = Session
                nIndex = nIndex + 1
            Next AppName
        End If

        Exit Sub

    myHandler:
        MsgBox Err.Description

    End Sub
```

Figure 3.11: GetApplicationSessionsAndPIDs Code Example

3.5.8 BeginContextChanges

3.5.8.1 Visual Basic Syntax

```
nContextToken = ContextManager.BeginContextChanges
(nParticipant)
```

Where:

nContextToken =	A long value containing the context ID for the data change transaction; required
ContextManager =	A valid context manager interface; required
nParticipant =	The participant ID of the application changing the data; required

RETURN VALUES

BeginContextChanges has five return values.

Table 3.6: Return Values for BeginContextChanges

HRResult	Description
S_OK	The call succeeded and has returned the expected value or values.
0x80000302 (2147484418): Invalid participant token	Participant token is not valid.
0x80000304 (2147484420) : Already in transaction	Only one transaction can be open per session. This is generated when a participant uses JoinContextSession with a value of False for bWaitForTransaction. Otherwise, the call blocks until the transaction is complete.

This page contains Eclipsys proprietary information and is not to be duplicated or disclosed to unauthorized persons.

Table 3.6: Return Values for BeginContextChanges

HRResult	Description
0x80000306 (2147484422) : Session is owned	A different participant owns this session.
0x80000308 (2147484424) : Session is frozen	The session is currently frozen.

REMARKS

BeginContextChanges creates a new set of context data and assigns it a unique Context Token. The server returns an error if another context change is currently in progress. When the server can create a new context, it sets a flag indicating that a context change is in progress, preventing other participants from changing the context.

If a participant has not completed a context change within a configurable time limit, the server attempts to Ping the participant using the callback interface. If Ping fails, the context changes are canceled. This is to prevent the server from being hung when a client dies. The server is allowed to cancel anyway.

If another client has a transaction open, the method blocks the client if the bWaitForTransaction parameter of JoinContextSession method was True. Otherwise, an error is returned.

SAMPLE CODE

```
Dim CMObject as New CoContextManager
Dim ifContextManager as IHVCContextManager
Dim ifParticipant as New CoParticipant
Set ifContextManager = CoContextManager
...
Dim nSession As Long
nSession = 65536 'This value comes from existing session
nParticipant = ifContextManager.JoinContextSession(nSession,
GetProcessID(), _
True, ifParticipant)
...
Dim nContextToken As Long
nContextToken = ifContextManager.BeginContextChanges(nParticipant)
```

Figure 3.12: BeginContextChanges Code Example

SEE ALSO

PublishContextChanges, TakeOwnership, FreezeContextSession

This page contains Eclipsys proprietary information and is not to be duplicated or disclosed to unauthorized persons.

3.5.9 PublishContextChanges

3.5.9.1 Visual Basic Syntax

```
ContextManager.PublishContextChanges nContextToken, bAccept
```

Where:

ContextManager = A valid context manager interface; required

nContextToken = A long value containing the context ID for the data change transaction; required

bAccept = A Boolean value indicating whether to publish the changes; required

RETURN VALUES

PublishContextChanges has five return values.

Table 3.7: Return Values for PublishContextChanges

HRResult	Description
S_OK	The call succeeded and has returned the expected value or values.
0x80000302 (2147484418): Invalid participant token	Participant token is not valid.
0x80000304 (2147484420) : Already in transaction	Only one transaction can be open per session. This is generated when a participant uses JoinContextSession with a value of False for bWaitForTransaction. Otherwise, the call blocks until the transaction is complete.
0x80000306 (2147484422) : Session is owned	A different participant owns this session.
0x80000308 (2147484424) : Session is frozen	The session is currently frozen.

REMARKS

PublishContextChanges causes the server to close the current context change transaction. If bAccept is True, the new context is made current and all the participants, except the initiating participant, joined to this context session have their ContextChanged methods called, passing the new context token. If bAccept is False, the context identified by nContextToken is cleaned up and the current change is closed. No notifications are sent. There is a 10-second window to complete transactions following a call to BeginContextChanges. If the time closes an open transaction, the error Not in Transaction is returned.

SAMPLE CODE

```
Dim CMObject as New CoContextManager
```


This page contains Eclipsys proprietary information and is not to be duplicated or disclosed to unauthorized persons.

```
Dim ifContextManager as IHVCContextManager
Dim ifParticipant as New CoParticipant
Set ifContextManager = CoContextManager
...
Dim nSession As Long
nSession = 65536 'This value comes from existing session
nParticipant = ifContextManager.JoinContextSession(nSession,
GetProcessID(), _
True, ifParticipant)
...
Dim nContextToken As Long
nContextToken = ifContextManager.BeginContextChanges(nParticipant)
... 'Change data here
ifContextManager.PublishContextChanges nContextToken, True
```

Figure 3.13: PublishContextChanges Code Example

SEE ALSO

BeginContextChanges

3.5.10 TakeOwnership

3.5.10.1 Visual Basic Syntax

```
ContextManager.TakeOwnership nParticipant
```

Where:

ContextManager = A valid context manager interface; required

nParticipant = A long value containing an existing participant ID; required

RETURN VALUES

TakeOwnership has four return values.

Table 3.8: Return Values for TakeOwnership

HRResult	Description
S_OK	The call succeeded and has returned the expected value(s).
0x80000302 (2147484418): Invalid participant token	Participant ID does not represent a valid participant in this session.
0x80000304 (2147484420): Already in Transaction	Cannot take ownership while an open transaction exists. Generated when a participant uses JoinContextSession with a value of False for bWaitForTransaction. Otherwise, the call blocks until the transaction is complete.
0x80000306 (2147484422): Session is owned	Only one owner allowed.

This page contains Eclipsys proprietary information and is not to be duplicated or disclosed to unauthorized persons.

REMARKS

This method prevents other applications within the context session from changing the data in the context session. Only the application with ownership can initiate changes. Participants who have previously joined the context session can call only this method.

SAMPLE CODE

```
Dim CMObject as New CoContextManager
Dim ifContextManager as IHVCContextManager
Dim ifParticipant as New CoParticipant
Set ifContextManager = CoContextManager
...
Dim nSession As Long
nSession = 65536 'This value comes from existing session
nParticipant = ifContextManager.JoinContextSession(nSession,
GetProcessID(), _
True, ifParticipant)
...
ifContextManager.TakeOwnership nParticipant
...
ifContextManager.ReleaseOwnership nParticipant
```

Figure 3.14: TakeOwnership Code Example

SEE ALSO

JoinContextSession, FreezeContextSession, ReleaseOwnership

3.5.11 ReleaseOwnership

3.5.11.1 Visual Basic Syntax

```
ContextManager.ReleaseOwnership nParticipant
```

Where:

ContextManager =	A valid context manager interface; required
nParticipant =	A long value containing an existing participant ID; required

This page contains Eclipsys proprietary information and is not to be duplicated or disclosed to unauthorized persons.

RETURN VALUES

ReleaseOwnership has four return values.

Table 3.9: Return Values for ReleaseOwnership

HRResult	Description
S_OK	The call succeeded and has returned the expected value or values.
0x80000302 (2147484418): Invalid participant token	Participant ID does not represent a valid participant in this session.
0x80000306 (2147484422): Session already owned	This session is owned by a different participant.
0x80000307 (2147484423): Session not owned	This session is not owned by any participant.

REMARKS

This method releases an owned context session. Only the application that has ownership can release the context session.

SAMPLE CODE

```
Dim CMObject as New CoContextManager
Dim ifContextManager as IHVCContextManager
Dim ifParticipant as New CoParticipant
Set ifContextManager = CoContextManager
...
Dim nSession As Long
nSession = 65536 'This value comes from existing session
nParticipant = ifContextManager.JoinContextSession(nSession,
GetProcessID(), _
True, ifParticipant)
...
ifContextManager.TakeOwnership nParticipant
...
ifContextManager.ReleaseOwnership nParticipant
```

Figure 3.15: ReleaseOwnership Code Example

SEE ALSO

TakeOwnership

3.5.12 FreezeContextSession

3.5.12.1 Visual Basic Syntax

```
ContextManager.FreezeContextSession nProcessID, nSessionToken
```

Where:

ContextManager = A valid context manager interface; required

This page contains Eclipsys proprietary information and is not to be duplicated or disclosed to unauthorized persons.

nProcessID = A long value identifying the process ID of the application freezing the context session; required

nSessionToken = A long value identifying an existing session; required

Return Values

FreezeContextSession has four return values.

Table 3.10: Return Values for FreezeContextSession

HRResult	Description
S_OK	The call succeeded and has returned the expected value or values.
0x80000302 (2147484418): Invalid session token	This session ID does not exist in the context manager.
0x80000313 (2147484435): Invalid process ID	The process ID does not exist in the process list for the workstation.
0x80000308 (2147484424): Session already frozen	This session has been frozen by another process.

Remarks

FreezeContextSession freezes content changes for the session. This prevents another application from changing the context for the session. Unlike TakeOwnership, an application does not need to be a participant to freeze the context session. The process identifier for the application is used to ensure that the application that froze the session is still operating. If another client has a transaction open, this method is blocked until the transaction is complete.

Sample Code

```
Dim CMObject as New CoContextManager
Dim ifContextManager as IHVCContextManager
Dim ifParticipant as New CoParticipant
Set ifContextManager = CoContextManager
...
ifContextManager.FreezeContextSession GetProcessID(), nSessionID
...
ifContextManager.ReleaseContextSession GetProcessID(), nSessionID
```

Figure 3.16: FreezeContextSession Code Example

See also

TakeOwnership, ReleaseSession

This page contains Eclipsys proprietary information and is not to be duplicated or disclosed to unauthorized persons.

3.5.13 ReleaseSession

3.5.13.1 Visual Basic Syntax

```
ContextManager.ReleaseSession nProcessID, nSessionToken
```

Where:

ContextManager =	A valid context manager interface; required
nProcessID =	A long value identifying the process ID of the application freezing the context session; required
nSessionToken =	A long value identifying an existing session; required

RETURN VALUES

ReleaseSession has five return values.

Table 3.11: Return Values for ReleaseSession

HRResult	Description
S_OK	The call succeeded and has returned the expected value or values.
0x80000302 (2147484418): Invalid session token	This session ID does not exist in the context manager.
0x80000313 (2147484435): Invalid process ID	The process ID does not exist in the process list for the workstation.
0x80000308 (2147484424): Session already frozen	This session has been frozen by another process.
0x80000309 (2147484425): Session not frozen	The session is not frozen by any process.

REMARKS

This method reverses the behavior of FreezeContextSession when an application wishes to allow context changes for the session. The process identifier is required to ensure that the application that initiated the frozen session is the one releasing it for changes.

SAMPLE CODE

```
Dim CMObject as New CoContextManager
Dim ifContextManager as IHVCContextManager
Dim ifParticipant as New CoParticipant
Set ifContextManager = CoContextManager
...
ifContextManager.FreezeContextSession GetProcessID(), nSessionID
...
ifContextManager.ReleaseContextSession GetProcessID(), nSessionID
```

Figure 3.17: ReleaseSession Code Example

This page contains Eclipsys proprietary information and is not to be duplicated or disclosed to unauthorized persons.

SEE ALSO

FreezeContextSession

3.5.14 IHVCContextData

Context data is composed of an ItemName and a ItemValue. An ItemName is stored as a String data type (BSTR) and an ItemValue is stored as a Variant (VARIANT). ItemNames are stored in upper case with white space removed.

```
interface IHVCContextData : IDispatch
{
HRESULT GetContextItems([in] long nContext,
 [out,retval] LPVARIANT pvtItemNames);
HRESULT GetContextValues([in] long nContext,
 [in] VARIANT vtItemNames,
 [out] LPVARIANT pvtItemValues);
HRESULT SetContextValues([in] long nContext,
 [in] VARIANT vtItemNames,
 [in] VARIANT vtItemValues);
};
```

Figure 3.18: IHVCContextData Code Example

3.5.15 GetContextItems

3.5.15.1 Visual Basic Syntax

```
vtItemNames = ContextData.GetContextData nContextID
```

Where:

vtItemNames =	A variant containing an array of item names associated with values in the context manager; required
ContextData =	A valid ContextData interface; required
nContextID =	A long value containing an existing context data set; required

RETURN VALUES

GetContextItems has three return values.

Table 3.12: Return Values for GetContextItems

HRResult	Description
S_OK	The call succeeded and has returned the expected value or values.
0x80000301 (2147484417): Invalid context token	Context ID does not identify an existing set of context data.
0x80000302 (2147484418): Invalid session token	This session ID does not identify an existing session.

This page contains Eclipsys proprietary information and is not to be duplicated or disclosed to unauthorized persons.

REMARKS

This method retrieves the names of the context data items. Each data item is stored as name-value pairs.

SAMPLE CODE

```
Dim CMObject as New CoContextManager
Dim ifContextManager as IHVCContextManager
Dim ifContextData as IHVCContextData
Set ifContextManager = CoContextManager
Set ifContextData = CoContextManager
...
vtItemNames = ifContextData.GetContextItems(nContext)
vtItemValues = ifContextData.GetContextValues(nContext,
vtItemNames)
...
vtItemValues[0] = "New Value"
SetContextValues nContext, vtItemNames, vtItemValues
```

Figure 3.19: GetContextItems Code Example

SEE ALSO

SetContextValues, GetContextValues

3.5.16 GetContextValues

3.5.16.1 Visual Basic Syntax

```
vtItemValues = ContextData.GetContextValues nContextID,
vtItemNames
```

Where:

VtItemValues =	A variant containing an array of item values associated with the names passed in vtItemNames; required
ContextData =	A valid ContextData interface; required
nContextID =	A long value containing an existing context data set; required
vtItemNames =	A variant containing an array of item names associated with values in the context manager; required

RETURN VALUES

GetContextValues has four return values.

Table 3.13: Return Values for GetContextValues

HRResult	Description
S_OK	The call succeeded and has returned the expected value or values.
0x80000301 (2147484417): Invalid context token	Context ID does not identify an existing set of context data.

This page contains Eclipsys proprietary information and is not to be duplicated or disclosed to unauthorized persons.

Table 3.13: Return Values for GetContextValues

HRResult	Description
0x80000302 (2147484418): Invalid session token	This session ID does not identify an existing session.
0x80000310 (2147484432): Invalid item name	The item name could not be found in the context data set.

REMARKS

This method retrieves the values of the context data items associated with the array of names passed in. Each data item is stored as name-value pairs.

SAMPLE CODE

```
Dim CMObject as New CoContextManager
Dim ifContextManager as IHVCContextManager
Dim ifContextData as IHVCContextData
Set ifContextManager = CoContextManager
Set ifContextData = CoContextManager
...
vtItemNames = ifContextData.GetContextItems(nContext)
vtItemValues = ifContextData.GetContextValues(nContext,
vtItemNames)
...
vtItemValues[0] = "New Value"
SetContextValues nContext, vtItemNames, vtItemValues
```

Figure 3.20: GetContextValues Code Example

SEE ALSO

GetContextItems, SetContextValues

3.5.17 SetContextValues

3.5.17.1 Visual Basic Syntax

```
ContextData.GetContextValues nContextID, vtItemNames, vtItemValues
```

Where:

ContextData =	A valid ContextData interface; required
nContextID =	A long value containing an existing context data set; required
vtItemNames =	A variant containing an array of item names associated with values in the context manager; required
VtItemValues =	A variant containing an array of item values associated with the names passed in vtItemNames; required

This page contains Eclipsys proprietary information and is not to be duplicated or disclosed to unauthorized persons.

Return Values

SetContextValues has six return values.

Table 3.14: Return Values for SetContextValues

HRResult	Description
S_OK	The call succeeded and has returned the expected value or values.
0x80000301 (2147484417): Invalid context token	Context ID does not identify an existing set of context data.
0x80000302 (2147484418): Invalid session token	This session ID does not identify an existing session.
0x80000305 (2147484421): Not in transaction	No transaction is currently open.
0x80000310 (2147484432): Invalid item name	The item name could not be found in the context data set.
0x80000312 (2147484434): Name and Value arrays differ in size	The number of items in vtItemNames does not match the number of items in vtItemValues.

Remarks

This method sets the new name-value pairs for the context data. This method must be called from within a transaction initiated by the calling application. The context token passed out by BeginContextChanges identifies the transaction and identifies the caller as the initiator. Each token is parsed into its constituent components and re-assembled so that we can guarantee clients see the data in a consistent format. Spaces are removed and the context data is translated to upper case. For example, Client.IDType = MRN.id is stored as CLIENT.IDTYPE=MRN.ID.

Sample Code

```
Dim CMObject as New CoContextManager
Dim ifContextManager as IHVCContextManager
Dim ifContextData as IHVCContextData
Set ifContextManager = CoContextManager
Set ifContextData = CoContextManager
...
vtItemNames = ifContextData.GetContextItems(nContext)
vtItemValues = ifContextData.GetContextValues(nContext,
vtItemNames)
...
vtItemValues[0] = "New Value"
SetContextValues nContext, vtItemNames, vtItemValues
```

Figure 3.21: SetContextValues Code Example

See also

GetContextItems, GetContextValues

This page contains Eclipsys proprietary information and is not to be duplicated or disclosed to unauthorized persons.

3.5.18 IHVCContextParticipant

This interface is implemented by the clients of the context server, and can be for dual or dispatch interfaces. The clients pass their IDispatch pointer into the context server. The server calls these methods when context changes are published.

```
interface IHVCContextParticipant : IDispatch
{
HRESULT ContextChanged([in] long nContext);
HRESULT Ping();
}
```

Figure 3.22: IHVCContextParticipant Code Example

3.5.19 Participant_ContextChanged(nContext)

REMARKS

This is a callback method provided by the client to the context manager server. When this method is called, the client uses the context identifier to retrieve the new data items from the context manager.

SAMPLE CODE

```
Dim ppt As New Participant
...
Sub ppt_ContextChanged (nContext As Long)
Dim vtItemNames, vtItemValues As Variant
 vtItemNames = ifContextData.GetContextItems(nContext)
 vtItemValues = ifContextData.GetContextValues(nContext,
 vtItemNames)
 ... ' Use the new data items to synchronize
End Sub
```

Figure 3.23: Participant_ContextChanged Code Example

SEE ALSO

GetContextItems, GetContextValues

3.5.20 Participant_Ping()

REMARKS

This is a callback method provided by the client to the context manager. This call is made to determine the liveliness of a participant. The context manager calls this method when the instigator of a transaction has not called the PublishContextChanges method within the transaction timer period.

SAMPLE CODE

```
Dim ppt As New Participant
...
Sub ppt_Ping()
' Don't really need to do anything here except pass thru
End Sub
```

Figure 3.24: Participant_Ping() Code Example

This page contains Eclipsys proprietary information and is not to be duplicated or disclosed to unauthorized persons.

SEE ALSO

BeginContextChanges, PublishContextChanges

3.6 Objects

All the COM objects in this design support the ISupportErrorInfo interface.

3.6.1 Class CoContextManager

This object manages the creation and connection of clients and sessions. This object is a singleton.

```
coclass CoContextManager
{
[default] interface
IHVCContextManager;
interface IHVCContextManager2;
interface IHVCContextData;
};
```

Figure 3.25: Class CoContextManager Code Example

3.6.2 Context Data

The table below shows the data SCM uses for context. It specifies if the data is to be published or subscribed, and the business case involved. An * (asterisk) indicates data fields SCM requires to subscribe.

Table 3.15: Sunrise Clinical Manager Context Data (1 of 3)

Data	SCM Publishes	SCM Subscribes
UserID * UserName	When the user logs on to SCM from the Logon window.	If the user changes, log off. (If SCM was launched through the command line, the Logon window does not appear.) **Note:** If SCM created the session, it takes ownership and prevents other participants from changing the context.

This page contains Eclipsys proprietary information and is not to be duplicated or disclosed to unauthorized persons.

Table 3.15: Sunrise Clinical Manager Context Data (2 of 3)

Data	SCM Publishes	SCM Subscribes
PatientID * VisitNumber *	When the current visit is changed. For example, a new Patient List is created by the user, a new visit is selected, or the user selects Next or Previous.	Displays the visit if new Patient ID or Visit Number is in the Patient List. If the patient is not in the list, a temporary list is created and the new visit appears if the ID is in the database. If the visit does not exist, or the user does not have access rights to view the visit, the message, "Visit not found for this list, or access denied" appears. **Note:** The PatientID is expected to be the default Client ID Type specified for the facility, and the VisitNumber is expected to be the default Visit ID Type.
PatientLastName PatientGivenName PatientMiddleName PatientLocationFacilityID PatientLocationUnit PatientLocationRoom PatientLocationBed	When the current visit is changed. For example, a new Patient List is created by the user, a new visit is selected, or the user selects Next or Previous.	
ProviderID	After a successful logon to SCM.	

This page contains Eclipsys proprietary information and is not to be duplicated or disclosed to unauthorized persons.

Table 3.15: Sunrise Clinical Manager Context Data (3 of 3)

Data	SCM Publishes	SCM Subscribes
SCMOrderEntry	TRUE or FALSE based on whether the Uses SCM flag checkbox is selected for the location group of the currently selected patient visit. If the flag checkbox is selected, the value is TRUE. A value of TRUE indicates that users can place orders for this patient at this location using SCM. A value of FALSE indicates SCM users are unable to place orders, or use order maintenance functions, such as Discontinue or Suspend, for this patient.	

This page contains Eclipsys proprietary information and is not to be duplicated or disclosed to unauthorized persons.

This page contains Eclipsys proprietary information and is not to be duplicated or disclosed to unauthorized persons.

CHAPTER 4: MEDIA VIEWER

Contents

4.1 Overview 61
4.2 Configuring Media Display 62
4.3 Using HL7 Messages 70
4.4 Creating a Media Viewer Server 72

4.1 Overview

This chapter describes how to configure your system to use Media Viewer and the servers provided with Sunrise Clinical Manager, and how to write your own server to access additional media types.

Media Viewer allows the linking of results and documents in SCM to external media (for example, X-rays, ECGs, recorded dictations, and so on) for the same patient and document or result context. To accomplish this, extra information is sent with inbound document (MDM) or result (ORU) HL7 messages. This extra information contains a code that describes the media and determines which viewer to use to display the media.

In SCM, a camera icon appears next to results and documents that have linked media.

SCM includes three media viewer servers:

- Media Link (HTTPDisplay): Displays media in a special Internet Explorer window that has no menus, buttons, or address bar. It is also a modal window, which means that while it is open, no other windows can get focus, so users cannot switch back to SCM. This assures that patient context is maintained.

  Eclipsys recommends using Media Link to show all media that can be opened in Internet Explorer. Media Link is also used to integrate with Sunrise KBDI, Eclipsys PACS product, to provide single-sign-on functionality.

- IMNetImageDisplay: Displays images using IMNet's Image Broker to retrieve and make image data appear from their image database or from disk.

This page contains Eclipsys proprietary information and is not to be duplicated or disclosed to unauthorized persons.

- HVCWinDisplay: Displays media files using a simple DOS path by letting Windows pick the viewing application using file extension (type) associations. It is essential that all client workstations have a file association for the extension used to name the linked media.

4.2 Configuring Media Display

Media Viewer allows you to show any multimedia file using a separate In-Proc COM server (DLL). The design of Media Viewer is flexible enough to allow viewing any type of file, as long as a server is created that performs the required steps to retrieve and display the media for a particular media type.

By using COM and a common interface, no code changes are required to the SCM application when adding more viewers. As long as the new viewer implements the IHVCMediaViewer interface and is properly registered on the workstation and in SCM, the application can access the viewer.

To configure multimedia display, these steps must be followed:

- Ensure that the server is registered with the client operating system.
- Ensure that the server is registered in SCM. If it is not, add the server to the Registered Media Viewers dictionary.
- Confirm that media files have the appropriate file associations.
- Verify the location of the media.
- Configure the HL7 interfaces.

4.2.1 Ensuring that the Server is Registered

Before any media server can be used, it must be registered on all necessary machines. For the three provided servers, HVCWinDisplay.dll and IMNetImageDisplay.dll, and Media Link (HTTPDisplay.dll) this should have already been performed by installing SCM. If designing a server, the system administrator at the site must register the server manually on each client computer.

◆ To determine if a server is registered

1. From the Start menu, select Run.
2. In the Open field, type regedt32 and click OK. The Registry Editor opens.
3. Select the group HKEY_CLASSES_ROOT and look for the name of the server. For the SCM standard servers, these are HTTPDisplay.HTTPDisplay, HVCWinDisplay.HVCWinDisplay and IMNetImageDisplay.IMNetImageDisplay.
4. Close the Registry Editor.

This page contains Eclipsys proprietary information and is not to be duplicated or disclosed to unauthorized persons.

◆ To register a server manually

1 From the Start menu, select Run.

2 In the Open field, type regsvr32 ServerName.dll and click OK. The Registry Editor opens.

3 Select the group HKEY_CLASSES_ROOT and confirm that the name of the server exists.

4 Close the Registry Editor.

After the server is registered on client machines, continue by adding the server to the Registered Media Viewers Dictionary.

4.2.2 Adding the Server to the Registered Media Viewers Dictionary

The Registered Media Viewers dictionary provides a cross-reference between the HL7 code used to specify the media type and the program used to make that media type appear. The Registered Media Viewers window allows the following:

- Activation and deactivation of viewers.
- Adding and editing viewers.

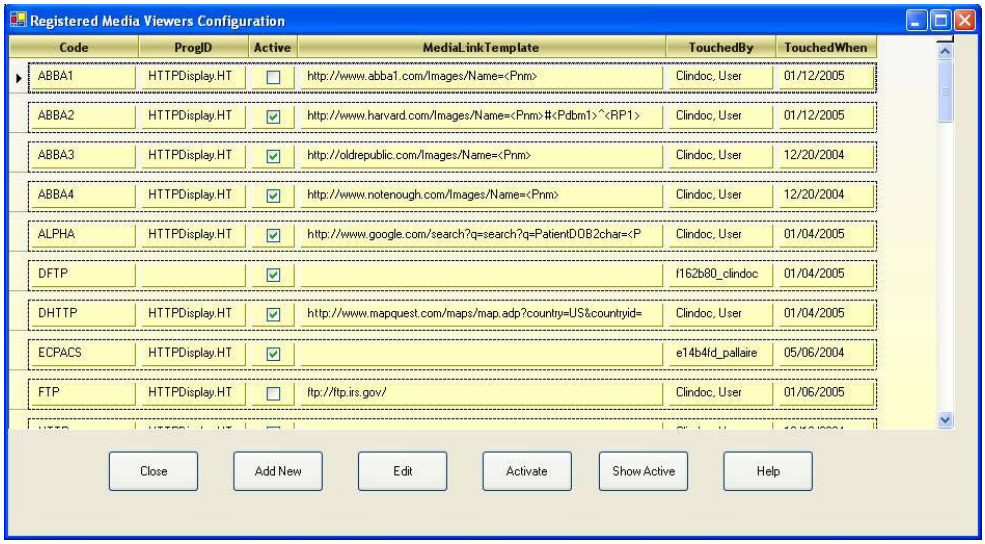


Figure 4.1: Registered Media Viewers Window

This page contains Eclipsys proprietary information and is not to be duplicated or disclosed to unauthorized persons.

Table 4.1: Registered Media Viewers Fields

Field	Description
Dictionary Code	The code used to specify the media type in HL7 messages. The maximum length is six characters.
Program ID	The ID used to identify the program. Must be in the format ProgramName.ServerName.Version. The version number is only necessary if the version being used is not the latest one.
Active	Shows the active viewers with a check mark.
Media Link Template	URL **Note:** The Media Link Template applies only to document viewer, the result viewer ignores this column.
Touched by	Login of the person that last edited the entry.
Touched when	Date of the last edit.

Note: If an HL7 transaction has been sent through the interfaces, a dictionary code can already exist for the viewer in the dictionary. It is automatically created by the HL7 interface if the system cannot find an appropriate code when the transaction is processed. If the interface automatically adds the dictionary code, the appropriate program ID must still be added for the media display to work.

◆ To enable or disable an existing media viewer

1 Select Configuration Tools >Dictionary >Miscellaneous > Registered Media Viewers to access the dictionary. The window in Figure 4.2 opens.

This page contains Eclipsys proprietary information and is not to be duplicated or disclosed to unauthorized persons.

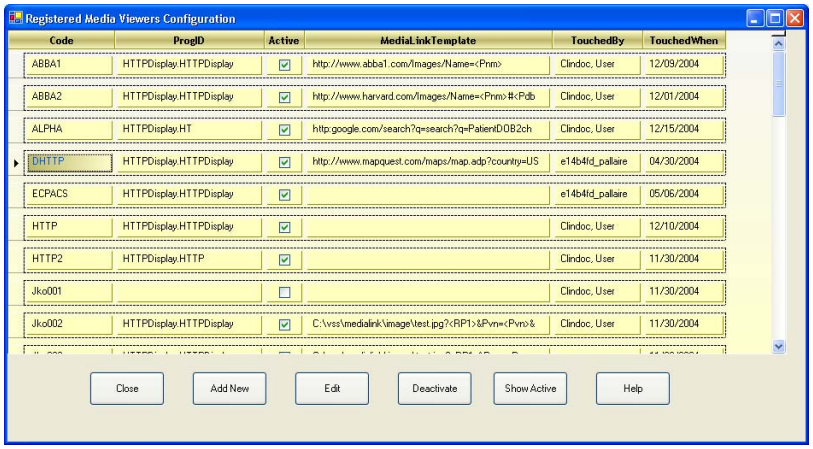


Figure 4.2: Registered Media Viewers Configuration Window

2 In the Activate column, check the box to enable the viewer.

To disable an entry, clear the check box in the activate column.

3 Click Close.

◆ To add or edit a media link (Documents viewer only)

1 Select Configuration Tools >Dictionary >Miscellaneous > Registered Media Viewers to access the dictionary and the window in Figure 4.2 opens.

2 Select an existing link and click Edit to make any changes. The Registered Media Viewers Edit/Add window opens and is shown in Figure 4.3 for an existing link.

If Add New is selected, the fields in the Registered Media Viewer Edit/Add window are blank.

This page contains Eclipsys proprietary information and is not to be duplicated or disclosed to unauthorized persons.

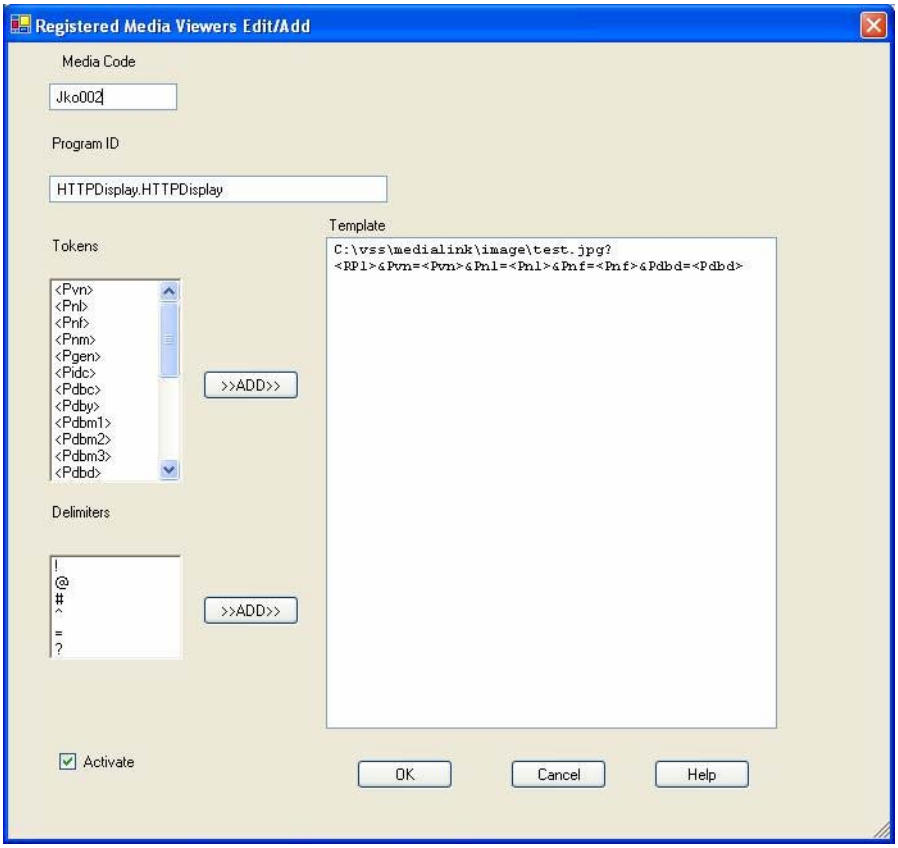


Figure 4.3: Registered Media Viewers Edit/Add Window

3 Type the Media Code and Program ID.

4 Use the Add buttons to add Tokens and Delimiters into the Template area as while constructing the new entry.

If editing a viewer, make edits directly in the Template area.

5 To enable an entry, select the Activate check box. Otherwise, the entry is added but not enabled.

6 To deactivate the selected entry, clear the Activate indicator.

7 Click OK to save the template.

After a template is entered, the template is checked for format. The user can accept the template or edit it. Examples of the checks performed are URL format and if all tokens entered are listed in the token list.

This page contains Eclipsys proprietary information and is not to be duplicated or disclosed to unauthorized persons.

4.2.2.1 Token List

Table 4.2 lists all the tokens and their definitions.

Table 4.2: Token List

Token	Definition	Token	Definition
<Pnl>	Patient current last name	<Pvn>	Patient visit number
<Pnm>	Patient current middle name	<Pnf>	Patient current first name
<Pidc>	Patient medical record number for currently selected visit	<Pgen>	Internal patient gender code
<Pdbc>	Patient DOB two-character century	<RPn>	The nth sub-component of the reference pointer
<Pdby>	Patient DOB two-character year	<Pdbm1>	Patient DOB two-character month
<Pdbd>	Patient DOB two-character day	<Pdbm2>	Patient DOB month full word. (JANUARY, FEBRUARY, MARCH)
<Pads1>	Patient street address	<Pdbm3>	Patient DOB month abbreviated word. (JAN, FEB, MAR)
<Padc>	Patient city	<Pads2>	Patient second street address
<Padz>	Patient ZIP code	<Pads>	Patient state
<DTs>	Current Date/time expressed in seconds that have elapsed since 1/1/1970	<xau>	Sunrise XA user id
		<DTh>	Current Date/time in the format: CCYYMMDDHHMMSS

If there is an image associated with the document, a camera icon appears next to the document on the Documents chart section Report view as shown in Figure 4.4.

This page contains Eclipsys proprietary information and is not to be duplicated or disclosed to unauthorized persons.

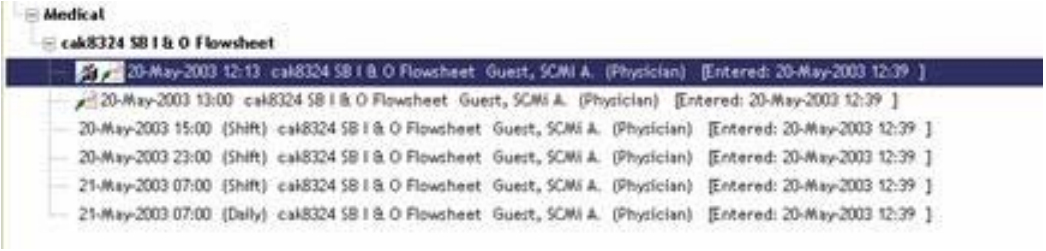


Figure 4.4: Camera Icon in Documents Chart Section

Note: Large images might impact load time of documents in the Document Chart section and structured notes.

4.2.3 Confirm Media File Type Associations

When using the HVCWinDisplay media viewer, it is essential that all client workstations have a file association for the extension used to name the linked media. For example, if the media appearing is drive:\path\filename.zzz, there must be an application associated with files of type zzz on client machines for that media to appear properly.

◆ To view or edit file type associations on a client machine

1 Open Windows Explorer.

2 From the Tools menu, select Folder Options. The Folder Options window opens.

3 Select the File Types tab. In the Registered file types list, view the file extensions and their associated application.

4 To edit a file type, select the file type and click Change. Select the program needed to open this type of file and click OK.

5 Click Close when done viewing or editing file types.

4.2.4 Verifying the Location of the Image Database

IMNetImageDisplay: Interfaces to IMNet Image Broker using their unique DOC ID to identify the image that is to be retrieved from their database and displayed. For testing purposes, locate the files on the computer's C:\ drive. To use a drive other than C:\, prefix a DOS path with the string `DOS:` For example, `DOS:\\graphics\images\image01.jpg`.

- HVCWinDisplay: Displays media files that are on the hard drive using a simple DOS path. Windows uses the file type associations to select the viewing application.
- Media Link: Only opens links that can be viewed in Internet Explorer. Verify that Internet Explorer can successfully open a sample link.

This page contains Eclipsys proprietary information and is not to be duplicated or disclosed to unauthorized persons.

4.2.5 Configuring Support for Sunrise PACS

If your facility uses Sunrise PACS, configure SCM and Sunrise PACS to use the same user names, and also configure a facility-specific system password.

4.2.5.1 Configuring Usernames

Configure usernames to be the same in SCM and Sunrise PACS. This should be considered during implementation, because it likely requires a manual process to manage the usernames. SCM does not support Master File Update messages for username maintenance. Configuring usernames in this way allows the Media Link viewer to authenticate users with Sunrise PACS when PACS images appear, thus providing single sign-on functionality.

4.2.5.2 Configuring the System Password

The system password is a facility-specific text string used by Sunrise PACS and SCM. It is stored in SCM in an environment profile. Create a new section called EclipsysPACS at the highest level of the environment profile, and then add an entry named KeyCode for the system password.

◆ To add a section to the environment profile

1 In the Configuration Tools window, click Dictionary Maintenance. The Dictionary Maintenance window opens.

2 In the Categories list, click Miscellaneous.

3 In the Dictionaries list, click Environment Profile.

4 Select any top-level section.

5 Click Add This Level. The Add Environment Profile window opens.

6 In the Profile Type field, select Section/Sub Section.

7 Enter a description for the section.

8 In the Code field, enter EclipsysPACS.

9 Click OK.

◆ To add an entry for the system password

1 Select the EclipsysPACS section.

2 Click Add Sub Level. The Add Environment Profile window opens.

3 In the Profile Type field, select Entry.

4 In the Code field, enter KeyCode.

5 In the Value field, enter the text needed to use as the system password.

This page contains Eclipsys proprietary information and is not to be duplicated or disclosed to unauthorized persons.

6 Click OK.

4.2.5.3 Configuring the HL7 Interfaces

Configure the interfaces send a new media type transaction. See the *Sunrise Clinical Manager HL7 Mapping Interface Reference Guide* for information. Sample messages are shown in the next section.

4.3 Using HL7 Messages

This section includes a sample HL7 message for sending an image.

The ImageType is ignored by our interface (other than being validated). Instead, we rely on the Media Code (also know as the Dictionary Code) and Access Token (ZIMNET, HARDDISKH0000200004e). When the SCM interfaces receive these messages, a MultimediaLink row is created in the database that contains the Media Code and Access Token. The MultimediaLink row is associated with the matching Result or Document that was also created. When SCM is required to display media associated with a result or document, the appropriate MultimediaLink row is retrieved, and the Media Code is cross-referenced with the Registered Media Viewers dictionary, which in turn specifies the Program ID of the viewer needed. This com server viewer is then activated and given the Access Token. It is now up to the COM server to retrieve and display whatever is referenced by the access token. Because this is an IMNET media code and Access Token, the IMNet Server (IMNetImageDisplay.dll) is started, which can retrieve and display images only.

Other media types can be viewed using the same HL7 message, except that the media code and Access Token have to be compatible with the viewer used to display the media.

Note: The results that are accessed using an URL need to specify the complete URL in the OBX-5 field as opposed to Documents that make use of the Media Link Template.

The following message creates a Skull or Sella Xray result, and attaches the image `C:\images\0\medic5.jpg` to it. To show the image, the viewer cross-referenced to ZHVCW (Eclipsys HVCWinDisplay) is used.

```
MSH|^~\&[|rad||PCS||1996052710500102||ORU^R01|0029%seq%|P|2.3|%seq%|
PID|||000009950^^^GVHADT|EXT000000001|JOHN^|PETERSON|19751205|M||C|2321 JOSE AVE^^Burnaby^BC^
V5A1N7 ^USA ^Home| |(604)555-2000||A|S|CHR||269492534|DL1000||B|||AMR|B
PV1||Inpatient|West^1C^Bed B^GVH|A|||vroe^Roe^ |apietri^Pietri^Angelo|ginnis^Innis ^Gladys|
MEDICINE/GENERAL|||||B6||jpeters^Peters^Jenny|Home|0002^^^Visit|||||||||||||||||||||||||1997010110
30|
OBR|||R8b^HCSLAB|SKULL OR SELLA XRAY^^LAB Standard|||199711100935||||||||||||||||F|||||
|||||
OBX|1|RP|SKULL OR SELLA XRAY^^LAB Standard||C:\E\images\E\0\E\medic5.jpg ^^Image&ZHVCW|||||F
NTE|||An image of a skull.|
```

Figure 4.5: Code Example

This page contains Eclipsys proprietary information and is not to be duplicated or disclosed to unauthorized persons.

The following message creates a document Routine Notes for patient John Peterson and attaches an image referenced with the DOC ID HARDDISKH0000200004e.

```
MSH|^~\&[|DOC|SCM|1996052710500102||MDM^T02|0029%seq%|P|2.3|%seq%|
EVN|T02|199605271050|
PID|||000009950^^^GVHADT|EXT000000001|JOHN^|PETERSON|19751205|M||C|2321 JOSE AVE^ ^Burnaby^
BC^V5A1N7^USA^Home||(604)555-2086||A|S|CHR||269492534|DL1000||B||||AMR|B|
PV1||Inpatient|West^1C^Bed B^GVH|A|||vroe^Roe^Victor |apietri^Pietri^Angelo |ginnis^Innis
^Gladys|MEDICINE/GENERAL|||||B6||jpeters^Peters^Jenny|Home|0002^^^Visit|||||||||||||||||||||||||19
9701011030|
TXA|001|Routine Notes^^LAB Standard||199702121130|||199702140750||jongmore^Ongmore^John^P ^^
Dr.|apietri^Pietri^^Angelo|jongmore^Ongmore^S^John^^Dr.|mdm1^FillFacID||002^PlacFacID|^HL7||Y||||T
XA 21: Change reason||
OBX|1|ST|Routine Notes^^LAB Standard||This is a standard text document sent via HL7. ||||||F
OBX|2|RP|Routine Notes^^LAB Standard||HARDDISKH0000200004e^^Image&ZIMNET||||||F
```

Figure 4.6: Code Example

The following message creates another Skull or Sella Xray result, and attaches a link to a Sunrise PACS image to it. To show the image, the Media Link viewer is used.

```
MSH|^~\&[|rad||PCS||1996052710500102||ORU^R01|0029%seq%|P|2.3|%seq%|
PID|||000009950^^^GVHADT|EXT000000001|JOHN^|PETERSON|19751205|M||C|2321 JOSE AVE^^Burnaby^BC^
V5A1N7 ^USA ^Home| |(604)555-2000||A|S|CHR||269492534|DL1000||B||||AMR|B
PV1||Inpatient|West^1C^Bed B^GVH|A|||vroe^Roe^ |apietri^Pietri^Angelo|ginnis^Innis ^Gladys|
MEDICINE/GENERAL|||||B6||jpeters^Peters^Jenny|Home|0002^^^Visit|||||||||||||||||||||||||1997010110
30|
OBR|||R8b^HCSLAB|SKULL OR SELLA XRAY^^LAB Standard|||199711100935|||||||||||||||||F|||||
|||||
OBX|1|RP|SKULL OR SELLA XRAY^^LAB
Standard||http://kbdi.hosp.edu?patient=000009950&order=R8b^^Image&ECPACS||||||F
NTE|||A KBDI image of a skull.|
```

Figure 4.7: Code Example

This page contains Eclipsys proprietary information and is not to be duplicated or disclosed to unauthorized persons.

The following message accesses a URL for the message template. The URL is located in the last line of the code example.

```
//DOC1: Simple message

MSH|^~\[|%send%||%receive%||%date%||MDM^T02|%seq%|P|2.3|%seq%|

EVN|T02|200101171410|

PID|||98354^^^ADT||QUILL

PV1||OUTPATIENT|^^^GVH|||||||||||||||2^^^Visit|||||||||||||||||||||||20010210

TXA|1|Nursing Note^Nursing
Note^Documents|FT|%date%%time%|||200102151210||cbannon^Bannon^Claire^^|||212629^DOC^^^|||212629TT^
DOC||DI

OBX|1|FT|Nursing Note^Nursing Note^Documents||Welcome to the CareVISION Document Receiver.||||||F|

OBX|1|FT|Nursing Note^Nursing Note^Documents||This document is a sample of receiving formatted
text.||||||F

OBX|1|FT|Nursing Note^Nursing Note^Documents||You will notice that each line of text Neuro: Pupils
4 mm bilaterally briskly reactive. Moves all four limbs to command. pt indicates she is unsteady on
feet due to visual deficiencies and intermittent numbness in lower limbs. Mobilizes on own with use
of a caneCVS:S1S2 distant muffled Moderate bilateral pitting edema of both ankles to mid-calf.
Unable to palpate pedal pulse due to gross edema. Apex and radial pulse 74/min and slightly
irregular. EGG shows SR with occasional PAC's. BP 100/80Resp: Pt is SOB at rest. She has difficulty
communication due to SOB. Symmetrical chest expansion. Air entry decreased to bases with coarse
crackles to mid-scapula. Pt states she normally is SOBOE but has had increasing SOB over the past
2-3 weeks. RP 28/min. Intermittent productive cough-slightly blood tinged frothy sputum. GI/GU:
Obese: GI/GU: Obese: presently following a 1000-calorie diabetic diet. Sugars are relativelystable;
no weight loss; Low urine output despi|| ||||F

OBX|1|FT|Nursing Note^Nursing Note^Documents||This is where the text for the document would
go!||||||F

OBX|2|RP|Nursing Note^Nursing
Note^Documents||E:\medialink\image\test.jpg?Pdbd=<Pdbd>&Pdbm3=<Pdbm3>&Pads1=<Pads1>&Pads2=<Pads2>&
Padc=<Padc>&Pads=<Pads>^[UID1[UIDType^Image[DHTTP[ZIMNET||||||F
```

Figure 4.8: Code Example

4.4 Creating a Media Viewer Server

4.4.1 Server Requirements

When writing a custom Media Viewer server, the following requirements must be met.

4.4.1.1 In-Proc Server

Define the media server as an In-Process server, or it is not loaded. The method CoCreateInstance() in SCM expects the server to be in this format, and does not search for any others.

4.4.1.2 Support IErrorInfo

The IErrorInfo interface is used to handle the passing of error messages back and forth between the server and the application. If the interface encounters an error, one of the three external methods exported by the IHVCMediaViewer interface should return an E_FAIL value and populate the IErrorInfo interface with the error message. See the IErrorInfo Implementation section at the end of this chapter to see how this can be implemented.

This page contains Eclipsys proprietary information and is not to be duplicated or disclosed to unauthorized persons.

4.4.1.3 Support IHVCMediaViewer

IHVCMediaViewer is the media server's primary interface. It must be implemented to be utilized by SCM. Follow exactly the interface specifications shown in this chapter.

4.4.2 IHVCMediaViewer Interface Specification

4.4.2.1 UUID

The IHVCMediaViewer interface must be defined to use the following UUID:

```
uuid(36868C0D-76FF-11D1-AA77-00A0244CFFCC)
```

This guarantees that SCM knows how to communicate with the server.

4.4.2.2 Methods

HRESULT Display([IN] BSTR pszToken, [IN] BSTR pszDescription)

This is the primary method used to display the image specified by the pszToken parameter. The second parameter, pszDescription, is often utilized as the title to the display window.

HRESULT CloseAll(void)

When this method is called, the server must close all the display windows that it opened. SCM calls this method whenever a patient is changed, or when the user logs off the system.

HRESULT Initialize(void)

This method is called just before the call to Display(). It can be utilized to perform any setup required before an image appears.

4.4.2.3 Sample Code

IDL Code

The following is a source IDL file that implements a simple server using the IHVCMediaViewer interface. The code in bold indicates the interface definition that must be the same for all servers.

```
// BASICTAB.IDL

import "oaidl.idl";
import "ocidl.idl";
// Forward defines
interface IPageViewSink;
[
    uuid(AC154CE2-27F9-11d3-A789-005004665F97),
    object,
    helpstring("IPageView interface"),
    oleautomation
]
interface IPageViewSink : IUnknown
{
    [helpstring("Retrieve the Patient Record names")]
```


This page contains Eclipsys proprietary information and is not to be duplicated or disclosed to unauthorized persons.

```
    HRESULT GetPatientRecordNames( [out,retval] SAFEARRAY(BSTR)*
Names );
    [helpstring("Retrieve the Patient Record values for the names
supplied")]
    HRESULT GetPatientRecordValues( [in] SAFEARRAY(BSTR)* Names,
                                    [out,retval] SAFEARRAY(BSTR)*
Values );
};

[
    uuid(AC154CE3-27F9-11d3-A789-005004665F97),
    object,
    helpstring("IPageViewSinkSet interface"),
]
interface IPageViewSinkSet : IUnknown
{
    [helpstring("Set the values")]
    HRESULT SetValues( LPOLESTR* pwcsNames,
                       LPOLESTR* pwcsValues,
                       long nValues );
};

[
    uuid(AC154CE0-27F9-11d3-A789-005004665F97),
    object,
    helpstring("IPageView interface"),
    oleautomation
]
interface IPageView : IUnknown
{
    // Base virtual functions
    [helpstring("Get the Menu Items")]
    HRESULT GetActionMenu( [out,retval] SAFEARRAY(BSTR)*
MenuPrompts );
    [helpstring("Menu item selected")]
    HRESULT FireActionMenuItem( [in] long MenuID );

    // Data Integrity
    [helpstring("Check if Patient can be changed")]
    HRESULT CanChangePatient();
    [helpstring("Check if Tab can be closed")]
    HRESULT CanCloseTab();

    // Navigation methods for tabbing
    [helpstring("Set focus to first control in tab")]
    HRESULT SetFocusFirst();
[helpstring("Set focus to last control in tab")]
    HRESULT SetFocusLast();
[helpstring("Is focus on first control in tab")]HRESULT
IsFirstWnd();
    [helpstring("Is focus on last control in tab")]
    HRESULT IsLastWnd();

    // Shell Info
    [helpstring("Add to event notification")]
    HRESULT AdviseSink( [in]IPageViewSink* PageViewSink,
                        [out,retval] long* ConnectId );
    [helpstring("Remove from event notification")]
    HRESULT UnadviseSink( [in] long ConnectId );
```


This page contains Eclipsys proprietary information and is not to be duplicated or disclosed to unauthorized persons.

```
    [helpstring("Called when Visit changes")]
    HRESULT OnChangeVisit();
};

[
    uuid(AC154CE1-27F9-11d3-A789-005004665F97),
    helpstring ("Sunrise Clinical Manager Basic Tab Interface
Library")
]
library BASICTABLib
{
    importlib("stdole2.tlb");
    importlib("stdole32.tlb");

    interface IPageView;
    interface IPageViewSink;
};
```

Figure 4.9: IHVCMediaViewer IDL Code Example

This page contains Eclipsys proprietary information and is not to be duplicated or disclosed to unauthorized persons.

This page contains Eclipsys proprietary information and is not to be duplicated or disclosed to unauthorized persons.

CHAPTER 5: DESKTOP INTEGRATION IN SUNRISE CLINICAL MANAGER

Contents

5.1 Overview 77
5.2 Common Terms and Phrases 78
5.3 Prerequisite Tasks 81
5.4 Modes of Context Sharing 81
5.5 Configuring the Vergence Context Manager for View Context Support 83
5.6 Configuring Sunrise Clinical Manager for Desktop Integration 86
5.7 Managing the Clinical Link 97
5.8 Responding to Context Changes 99

5.1 Overview

Desktop integration shares context data on the desktop, synchronizing two applications so a user can manage a patient's care in both applications without having to choose the same patient in both. HL7 Desktop Integration is a way for Sunrise Clinical Manager to share clinical and user context data with other third party or Eclipsys products, such as Sunrise Critical Care. The HL7 distinction means that the HL7 CCOW standard is followed.

Note: HL7 Desktop Integration is the preferred way of sharing context available in SCM. Another method, which existed prior to this release, uses a proprietary method of sharing the context. This proprietary method may not be supported in subsequent releases.

SCM can share context in one of two modes: peer or master. In peer mode, SCM sends and receives patient, user, and view context changes. In master mode, SCM sends patient, user and view changes, but only receives view changes.

This chapter summarizes the steps needed to set up and use SCM as a context participant for HL7 desktop integration.

This page contains Eclipsys proprietary information and is not to be duplicated or disclosed to unauthorized persons.

5.2 Common Terms and Phrases

The following terms are used throughout this chapter.

5.2.0.1 Automation Controller Application

An application that starts SCM using automation. Automation is a method whereby another application (called the controller) starts SCM, populates the Tools menu, logs a user on and off, changes a password, and controls the hiding and showing of SCM on the desktop. (See "Automation Control Interface," for more information on these features.)

SCM must be started using automation for it to participate in context sharing. If the user launches SCM directly, SCM operates in the normal stand alone mode only; and none of the desktop integration features described in this chapter are enabled.

Eclipsys provides the SCM Context Controller. This sample application uses automation to start SCM and set a number of parameters that control how SCM behaves. SCM Context Controller works only with SCM; it cannot be used to log on to any other Eclipsys applications and is included as part of the SCM Client. For more information on configuring and using SCM Context Controller, see "Configuring Sunrise Clinical Manager Context Controller" in this chapter.

Note: Write your own controller application in any computer language that supports automation. Scripting languages are not supported. If not using the SCM Context Controller, see "Automation Control Interface" for information on how to create a custom controller application.

5.2.0.2 Clinical Link

The term used to describe the state when two applications share the same context. The clinical link can be in one of three states.

- On: The applications are sharing context.
- Broken: Communication with the context sharing applications has stopped for various reasons.
- Changing: The view context is changing as the user selects a form or button from either configured application.

5.2.0.3 Common Context

The collection of data items and their values currently stored in the Context Manager. These data items can be used by any participating application to synchronize itself with other applications participating in desktop integration. The SCM integration process can use and maintain four data items in the context manager: patient MRN, patient visit number, user logon ID, and view name.

This page contains Eclipsys proprietary information and is not to be duplicated or disclosed to unauthorized persons.

5.2.0.4 Context

A generic term that refers to the value of various attributes in the application's environment. Typically, this means information that identifies the current patient and the current user.

5.2.0.5 Context Manager

The central repository of context data shared among all participating applications on the desktop. It is not part of SCM, and it must be CCOW-compliant. The Context Manager is the application that holds the current values of the common context and provides a method of communicating to the participating applications any changes to this information.

5.2.0.6 Context Participant Interface Object (CPIO)

The tool that SCM and other Eclipsys applications, such as Critical Care, use to communicate with the Context Manager. CPIO is responsible for the context management of user, patient, and custom subject areas using an interface to a CCOW-compliant context manager. The key custom subject areas supported by CPIO are the ability to set the view and the visit ID.

5.2.0.7 Current Clinical Context

Changing current clinical context involves changing one or more of these variables: the active application, the user logged in, or the selected patient. This is the context from the SCM perspective, and may or may not be the same as the common context. Because it is possible that SCM's clinical link can be broken, the patient as appearing in SCM no longer is the same as the patient stored in the common context.

5.2.0.8 Desktop Integration

The sharing of context data on the desktop that allows two applications to synchronize the patient medical record number, patient visit number, user login ID and view name, allowing both applications to handle the same user and patient automatically.

5.2.0.9 Master Context Participant

This indicates the controlling application when one application starts and controls another. The master application typically controls when the slave application appears. This is usually initiated by a button or menu item in the master application that invokes the slave application.

The master context participant controls the user context (user name and password), patient context, and view context. Slave applications can publish only a view context change to the master; all other types of context changes are ignored.

This page contains Eclipsys proprietary information and is not to be duplicated or disclosed to unauthorized persons.

5.2.0.10 Participating Application

This indicates a user application that is sharing the common context with other user applications.

SCM and Sunrise Critical Care are considered participating applications if users are involved in context sharing.

5.2.0.11 Participating in Context

Making a physical connection to the Context Manager so that SCM can publish context data or subscribe to context data set by another application.

5.2.0.12 Patient Context

This indicates the currently selected patient. This is typically represented in the common context by the patient's Medical Record Number (MRN) and visit number (if applicable).

5.2.0.13 Peer-to-Peer Context Participation

All participating applications are able to set the context.

5.2.0.14 Slave Context Participant

This indicates an application being controlled by and receiving common context data published by a master application. The slave application is passive; it cannot control the context.

5.2.0.15 User Context

The currently logged on user represented by the user's logon ID and password.

5.2.0.16 View Context/View Name

This indicates the currently active application and the function being performed. The view context tells an application where the user should be positioned when the application is started.

View context is available only with selected Eclipsys applications, and the valid view names are determined by each application.

This page contains Eclipsys proprietary information and is not to be duplicated or disclosed to unauthorized persons.

5.3 Prerequisite Tasks

For SCM to participate in context sharing, the tasks listed in Table 5.1 must be performed:

Table 5.1: Task List

Prerequisite Task	Refer to
1. Install the SCM Client workstation.	Chapter 6, “Installing Workstations” in the *Sunrise Clinical Manager Installation Guide.*
2. Install the other applications that are to share context.	This can be Sunrise Critical Care, or whichever healthcare application that SCM needs to share clinical context data with. These other applications must also be able to communicate with a CCOW-compliant context manager for HL7 desktop integration to work. See the installation guide that came with the software for the participating application.
3. Install the Context Manager.	Follow instructions that are provided with the Context Manager software. If using the Sentillion Vergence Context Manager and you wish to exchange View context, configure the Context Manager to support view context; see “Configuring the Vergence Context Manager for View Context Support” later in this chapter.
4. Install the Context Participant Interface Object (CPIO) package.	To install this, run Setup.exe from the SCM CD and select ‘Sunrise Desktop Integration’ from the install menu.
5. Install and configure the automation controller application.	For SCM Context Controller, see “Configuring Sunrise Clinical Manager Context Controller” on page 96.

5.4 Modes of Context Sharing

SCM can share patient context information with other clinical applications running on the same desktop, such as Sunrise Critical Care. SCM participates in context only if it is started using automation, which can be called by the the SCM Context Controller application, or a custom-written controller application that is created. For more information on creating a custom controller application, see “Automation Control Interface.”

SCM can participate in context in two modes peer-to-peer or master. For each mode, the context may or may not include a visit ID. According to the HL7 standard, a common vocabulary for context has been defined. The context is referred to as the "clinical link."

This page contains Eclipsys proprietary information and is not to be duplicated or disclosed to unauthorized persons.

5.4.1 Peer-to-Peer

In peer-to-peer mode, the context participants act as peers, meaning that any application can change the current clinical context. This includes setting and receiving user context.

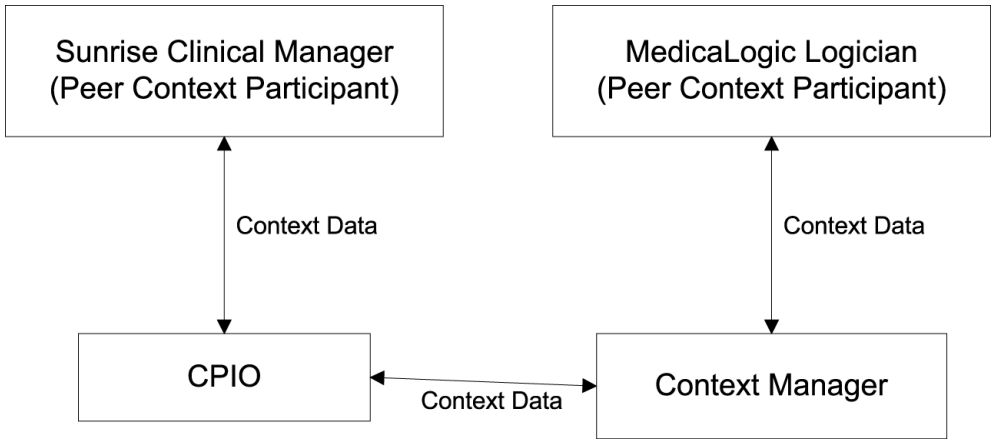


Figure 5.1: Peer-to-Peer Diagram

5.4.2 Master

SCM acts as the master application that controls the context. The user can only change the view context in SCM from another application. SCM ignores any user or patient context changes. In this mode, SCM starts another application as a slave application using an entry in the Tools menu.

5.4.3 Setting the Participation Mode

SCM knows which type of context participation is expected, based on the parameters provided by the application that starts SCM using automation. Create your own custom context controller application (see "Automation Control Interface") or use the supplied sample controller application (scmcc.exe).

The application that starts SCM must provide parameters to indicate the mode in which SCM is to be started (peer-to-peer or master), as well as how to treat unsaved data when the context is changed. For information on specifying the parameters for SCM Context Controller, see "Configuring Sunrise Clinical Manager Context Controller."

The following are assumptions for dealing with a single sign-on application:

- SCM is started using automation.
- Problems with logon are sent as returned values using automation.

This page contains Eclipsys proprietary information and is not to be duplicated or disclosed to unauthorized persons.

- The single sign-on application does not allow a user to log off until SCM has been logged out. If there is unsaved data and the AllowDataLoss flag is TRUE, SCM logs off and causes all unsaved data to be lost. If the AllowDataLoss flag is FALSE, then SCM displays a window indicating that there is still unsaved data and does not shut down until an action has been taken. The single sign-on application must remain logged on until action has been taken.
- SCM is configured to prevent Suspend and AutoLogoff from within SCM. For more information see the *Sunrise Clinical Manager General Configuration Guide*.

5.5 Configuring the Vergence Context Manager for View Context Support

The default installation of the Vergence Context Manager does not support the exchange of view context data between participating applications. Therefore, if exchanging View context is required, after you install the context manager, it must be configured.

◆ To configure the Vergence Context Manager

1 Install the Vergence Desktop Components application.

2 Using the Web Browser, connect to the Vergence Vault using the URL given by Sentillion. A login screen opens.

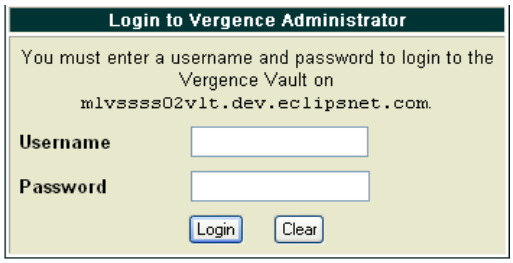


Figure 5.2: Vergence Vault Login Screen

3 Select the Vergence Configuration Tab to begin.

This page contains Eclipsys proprietary information and is not to be duplicated or disclosed to unauthorized persons.

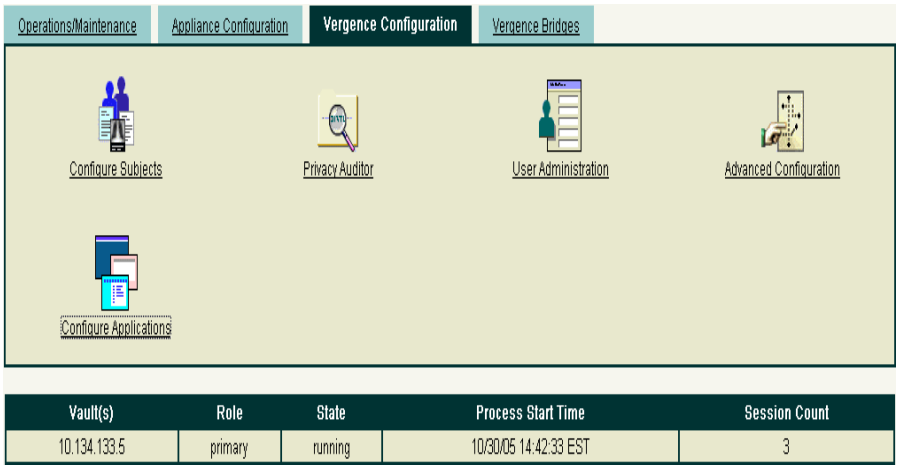


Figure 5.3: Vergence Configuration Tab

4 Select Configure Applications.

5 From this listing it is possible to edit an existing application or to create one if the Applications list does not include an Application entry, using the following steps:

a Click Add to add a new application. The Add Application window appears as shown in Figure 5.4.

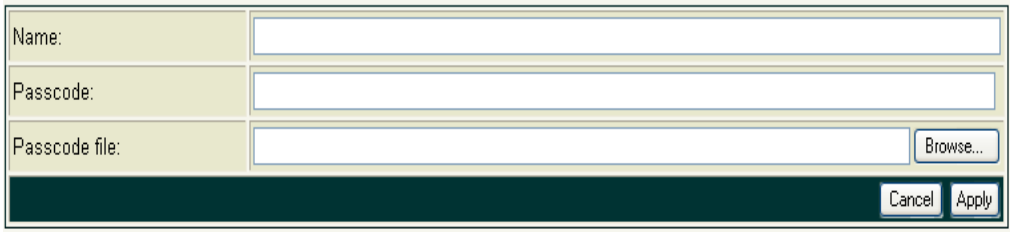


Figure 5.4: Add Application Window

b In the Application Name field, enter the name of the application.

Note: Use the exact name of the application, as set in the CVApplicationName environment profile.

c In the Passcode field, enter the passcode that is associated with the application.

Note: Use the exact Passcode that applies to the application being added. The Desktop Vault uses Passcode to authenticate the application to participate in Context Management. The Passcode is encrypted and decrypted by the Desktop Vault for security purposes.

This page contains Eclipsys proprietary information and is not to be duplicated or disclosed to unauthorized persons.

6 After adding applications is finished, click Configure Subjects from the main Vergence Configuration screen.

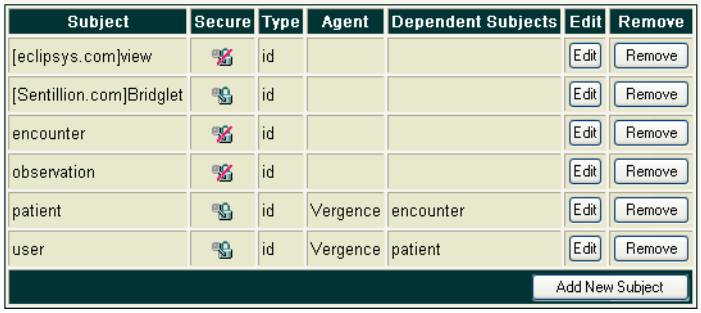


Figure 5.5: Modify Site Configuration Window

If the Subjects list does not include a View entry, create one using the following steps:

a Click Add New Subject. The Add New Subject Name window opens and is shown in Figure 5.6.

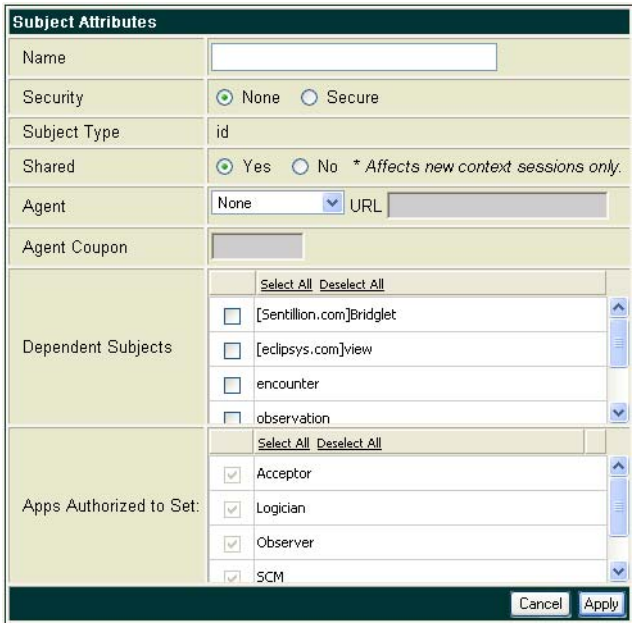


Figure 5.6: Add New Subject Window

b Enter `[eclipsys.com]View` and in the Subjects list, select [eclipsys.com]View. The Items list on the right side of the window should be empty.

c Click Apply. At a minimum, View, Encounter, Patient and User subjects should be entered.

This page contains Eclipsys proprietary information and is not to be duplicated or disclosed to unauthorized persons.

7 Proceed with the next prerequisite task of installing and configuring the Context Participant Interface Object. (See step 4 in "Prerequisite Tasks," earlier in this chapter.)

5.6 Configuring Sunrise Clinical Manager for Desktop Integration

After having performed the prerequisite installation and configuration tasks, it is time to configure SCM to participate in context sharing on the desktop.

5.6.1 Configuring View Change Requests

View change requests are generated when the user selects a specially configured menu option in the SCM Tools menu. SCM parses the launch information and executes the command line. When participating in context, the Tools menu options should be set using the automation controller application (see "Automation Control Interface"). This method shows the Tools options only when SCM is participating in context sharing.

5.6.2 Setting the Tools Menu Options Using the Environment Profile

Set the Tools menu options in the environment profile, as described in table Table 5.2. However, once done, the menu options are visible in SCM whether or not it is participating in context sharing.

For each entry added, increase the HVCAppCount value.

Set the values in the Environment Profile Maintenance window (Configuration Tools > Dictionary icon > Dictionary Maintenance window > Miscellaneous category > Environment Profile dictionary).

Table 5.2: Value List (1 of 2)

Section>Entry	Default Value	Description
HVC Global Workstation>Tools>HVC AppCount	0	The total number of applications present in the Tools menu.
HVC Global Workstation>Tools>HVC App0		Name and path of the first application in the Tools menu.
HVC Global Workstation>Tools>HVC App1		Name and path of the second application in the Tools menu.
HVC Global Workstation>Tools>HVC App2		Name and path of the third application in the Tools menu.

This page contains Eclipsys proprietary information and is not to be duplicated or disclosed to unauthorized persons.

Table 5.2: Value List (2 of 2)

Section>Entry	Default Value	Description
HVC Global Workstation>Tools>HVC App3		Name and path of the fourth application in the Tools menu.
HVC Global Workstation Tools>HVCApp4		Name and path of the fifth application in the Tools menu.
HVC Global Workstation>Tools>HVC App5		Name and path of the sixth application in the Tools menu.
HVC Global Workstation>Tools>HVC App6		Name and path of the seventh application in the Tools menu.
HVC Global Workstation>Tools>HVC App7		Name and path of the eighth application in the Tools menu.
HVC Global Workstation>Tools>HVC App8		Name and path of the ninth application in the Tools menu.
HVC Global Workstation>Tools>HVC App9		Name and path of the tenth application in the Tools menu.

5.6.3 Adding Selections to the Tools Menu in Sunrise Clinical Manager

The following procedure uses the environment profile entries to populate the Tools menu in SCM. However, as described earlier, the preferred method is to use the Context Controller to do this.

◆ To add selections to the Tools menu

1 In the Configuration Tools window, double-click Dictionary. The Dictionary Maintenance window opens and is shown in Figure 5.7.

This page contains Eclipsys proprietary information and is not to be duplicated or disclosed to unauthorized persons.

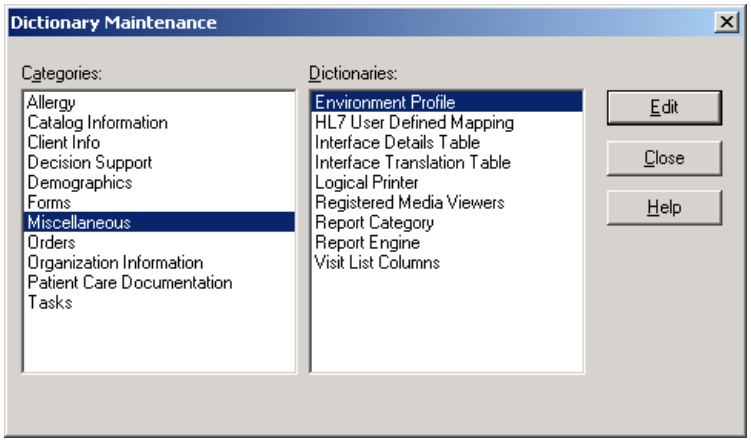


Figure 5.7: Dictionary Maintenance Window

2 In the Categories list, select Miscellaneous.

3 In the Dictionaries list, double-click Environment Profile. The Environment Profile Maintenance dictionary opens as shown in Figure 5.8.

4 Click the plus sign next to the HVC Global Workstation section to view its subsections.

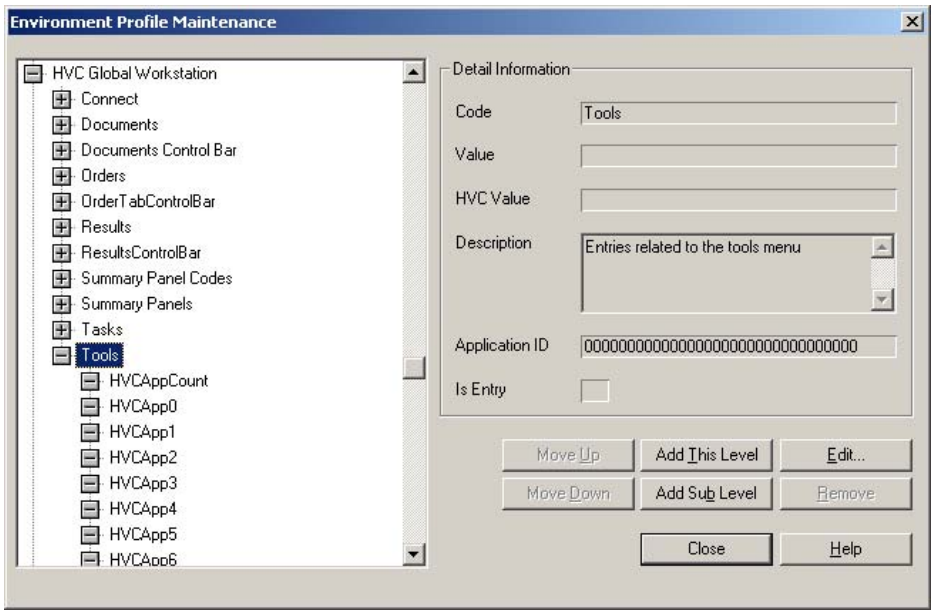


Figure 5.8: Environment Profile Maintenance Window

5 Click the plus sign next to Tools to view its entries.

6 Select the next available application slot where to add the tool (for example, HVCApp0) and click Edit. The Edit Environment Profile window opens as shown in Figure 5.9.

This page contains Eclipsys proprietary information and is not to be duplicated or disclosed to unauthorized persons.

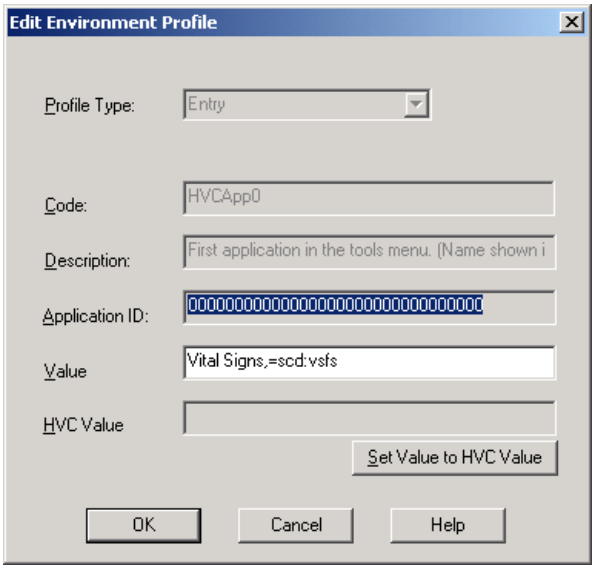


Figure 5.9: Edit Environment Profile Window

7 In the Value field, enter the name to appear in the Tools menu (for example, Vital Signs), followed by a comma. Next, enter an equal sign (=) followed by the command line specifying the view (for example, scd:vsfs). Continue to the next section, "Configuring Tools Menu Actions" for more information.

8 Click OK. Return to the Environment Profile Maintenance window.

9 Select HVCAPPCount and click Edit. The Edit Environment Profile window opens as shown in Figure 5.10.

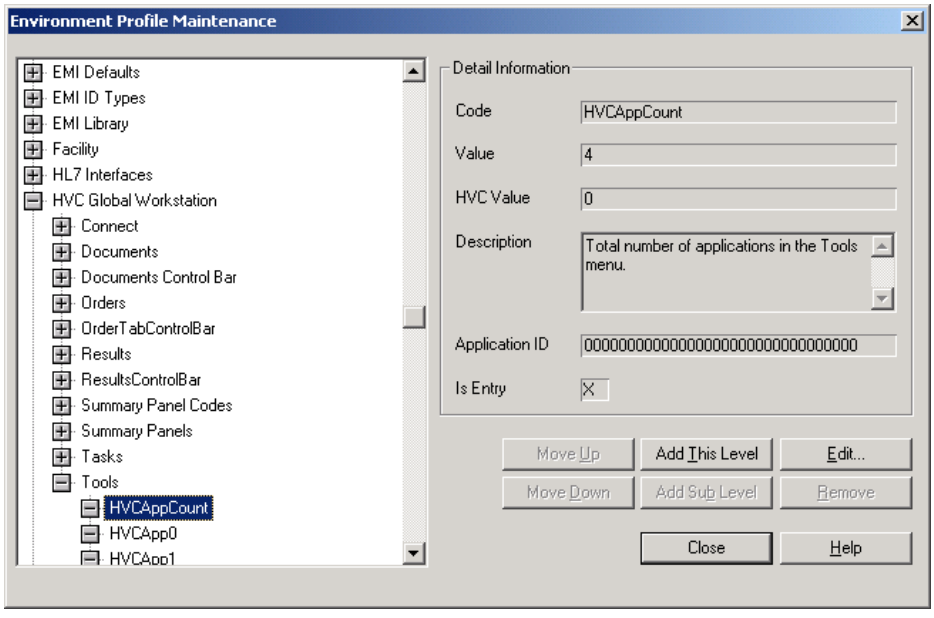


Figure 5.10: Environment Profile Maintenance Window

This page contains Eclipsys proprietary information and is not to be duplicated or disclosed to unauthorized persons.

10 In the Value field, add 1 to the number listed, indicating the total number of entries in the Tools menu as shown in Figure 5.11.

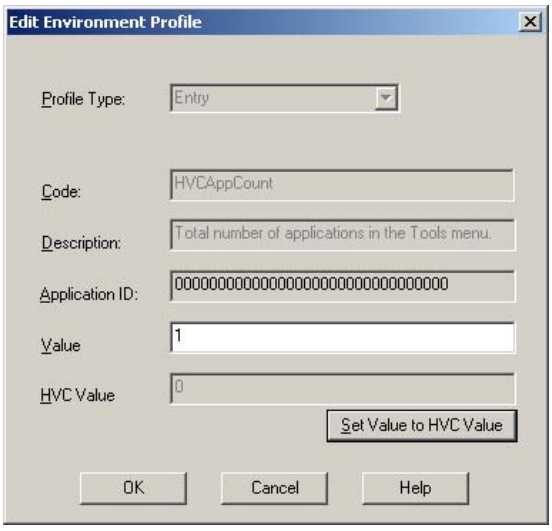


Figure 5.11: Edit Environment Profile Window

11 Click OK.

12 Click Close.

5.6.4 Configuring Tools Menu Actions

To add an item to the Tools menu, enter the name to appear in the Tools menu, followed by a comma, then the command string. To change the view context, enter an equal sign before the command string. Any command line preceded by an equal sign is interpreted as a view name to send to the HL7 desktop.

To launch another program:

```
Menu_Item_Name, Command_Line
```

Where:

Menu_Item_Name = the name appearing on the Tools menu

Command_Line = the application launched

For example:

```
Notepad, Notepad.exe c:\myfile.txt
```

To change the view context (in an application that supports this feature):

```
Menu_Item_Name, =View_Name
```


This page contains Eclipsys proprietary information and is not to be duplicated or disclosed to unauthorized persons.

Where:

Menu_Item_Name =	the name appearing on the Tools menu
View_Name =	the application launched and the location in that application

For example:

```
SCC, =Census
```

5.6.5 Enabling a Shortcut Key

To enable a shortcut key for the tool (allow access through keyboard commands), add an ampersand in front of the character to be used for the shortcut. For example, to use the letter S as the shortcut for the Sunrise Critical Care in the Tools menu, enter &SCC. To use the letter C as the shortcut for the Sunrise Critical Care in the Tools menu, enter S&CC. For example, the entire shortcut sequence would be Alt+T (to display the Tools menu), and then S or C to run the tool.

Add tools to the SCM toolbar using the Customize Toolbar window in SCM. To add this to all users' workstations, add the tool to SCM, then use that user profile when creating other users. One caveat is that if more than one tool is defined, each toolbar button displays the same wrench icon. To see what a tool is for, move the cursor over the button to display the tool tip.

WARNING



The wrench icon can be configured as a different icon, but the functionality of desktop integration is impaired by doing so. Icon configuration requires the use of a third-party resource editor software program.

5.6.6 Configuring Medical Record Numbers (MRNs)

The CPIOContextMRNType and CPIOContextMRNSite environment profile entries must be edited. This is necessary for the correct exchange of patient context data between SCM and CPIO.

SCM supports multiple MRNs for a patient. Each MRN has a type associated with it. The type is constant for a particular facility within the enterprise. For example, a patient might have MRN 1234 at facility A and MRN 3456 at facility B within the same enterprise.

SCM is limited to only one MRN type when accessed using CPIO and context for each facility.

An environment profile entry must be set to specify the MRN type to be used for matching MRN numbers in the context. The default entry is set in the HVCGlobalWorkstation section. This value needs to match the value set up in the Client ID Type under the Facility window. See “Setting Up Facilities” in Chapter 4, “Setting up the Enterprise Structure” in the *Sunrise Clinical Manager General Configuration Guide*. All patients and visits are found using the Client ID type and

This page contains Eclipsys proprietary information and is not to be duplicated or disclosed to unauthorized persons.

Visit ID type set up for the facility. The facility is determined by the location of the workstation.

◆ To configure MRN information for the Context Manager

1 In the Configuration Tools window, double-click Dictionary. The Dictionary Maintenance window opens.

2 In the Categories list, select Miscellaneous.

3 In the Dictionaries list, double-click Environment Profile. The Environment Profile Maintenance dictionary opens as shown in Figure 5.12.

4 Click the plus sign next to the HVC Global Workstation section to view its subsections.

5 Click the plus sign next to Connect to view its entries.

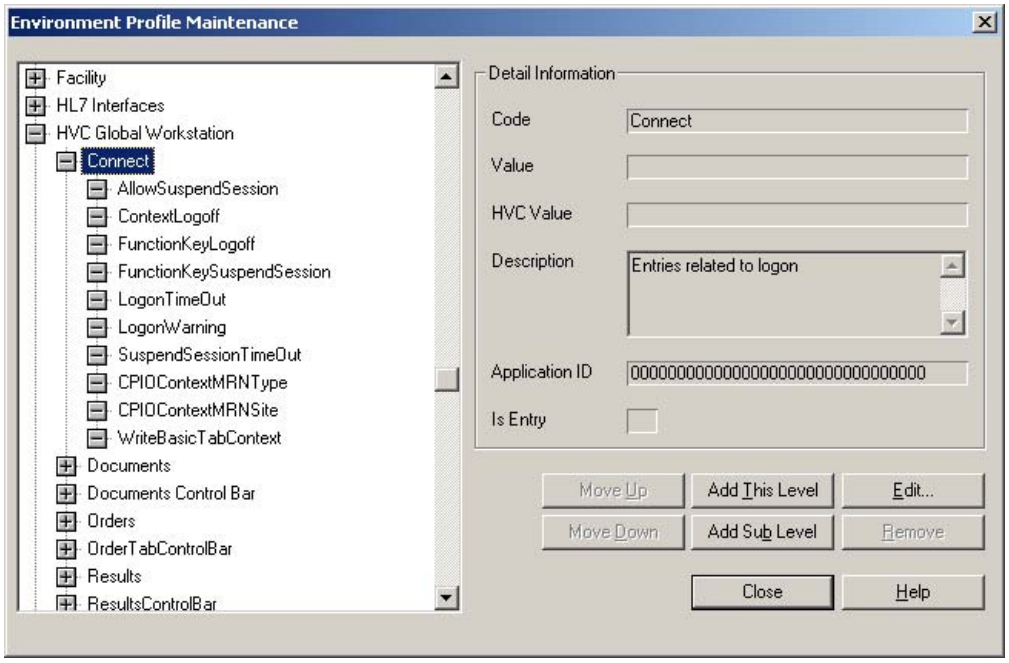


Figure 5.12: Environment Profile Maintenance Window

6 Select CPIOContextMRNType.

7 Click Edit. The Edit Environment Profile window opens as shown in Figure 5.13.

This page contains Eclipsys proprietary information and is not to be duplicated or disclosed to unauthorized persons.

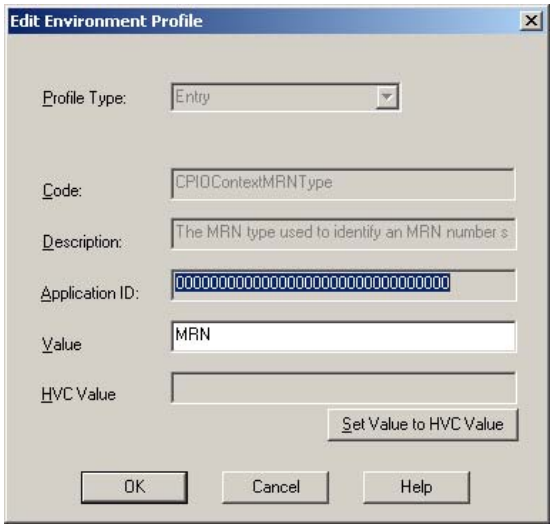


Figure 5.13: Edit Environment Profile Window

8 In the Value field, enter the MRN ID type used to identify MRN numbers supplied by the Context Manager. It must be a valid MRN type in the SCM Client ID Type dictionary.

9 Click OK. Return to the Environment Profile Maintenance window.

10 Select CPIOContextMRNSite and click Edit. The Edit Environment Profile window opens as shown in Figure 5.14.

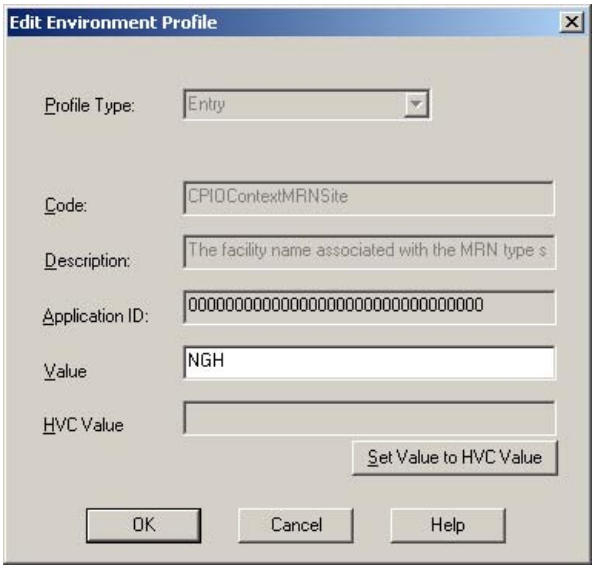


Figure 5.14: Edit Environment Profile Window

11 In the Value field, enter a site identifier for the MRN. While not required, it is recommended to use the facility name associated with the MRN type specified

This page contains Eclipsys proprietary information and is not to be duplicated or disclosed to unauthorized persons.

in the CPIOContextMRNType entry. All participating applications require that the same identifier be used.

Note: This is a CPIO requirement. Context data is transferred in name/value pairs. To retrieve the MRN from the Context Manager, SCM must also specify the site name. When SCM publishes a patient change, it sends out the new patient MRN. For example, if the CPIOContextMRNSite environment profile entry is set to Hospital A, it gets appended to the PATIENT.ID.MRN item name, as follows:

PATIENT.ID.MRN.HOSPITAL A

At the receiving end, the application looks for a data item named PATIENT.ID.MRN.HOSPITAL A and extracts its value. This is why it is imperative that the site name match across all applications. Otherwise, one application might publish the MRN as PATIENT.ID.MRN.HOSPITAL B, while another application is looking for PATIENT.ID.MRN.HOSPITAL A.

12 Click OK. Return to the Environment Profile Maintenance window.

13 Click Close.

5.6.7 Configuring Exit of all Participant Applications

SCM allows the option of exiting from all participant applications when SCM is exited in peer-to-peer mode. There are three available options:

- Exit from all applications.
- Exit from SCM only.
- A window appears giving the user the option of exiting from all applications or just SCM.

Set an environment profile entry to specify which option is desired. The default entry is to display the option window when the user logs off. This default entry is set in the HVCGlobalWorkstation section, but the enterprise can set up individual entries in the workstation profile should different workstations have different requirements.

If SCM automatically logs off, the behavior is per the SCM option. That is, SCM logs off but does not clear the user context, which would automatically log off other applications. The context controller clears the user context after all applications have exited context.

◆ To configure the exit option

1 In the Configuration Tools window, double-click Dictionary. The Dictionary Maintenance window opens.

2 In the Categories list, choose Miscellaneous.

This page contains Eclipsys proprietary information and is not to be duplicated or disclosed to unauthorized persons.

3 In the Dictionaries list, double-click Environment Profile. The Environment Profile Maintenance window opens.

4 Click the plus sign next to the HVC Global Workstation section to view its subsections.

5 Click the plus sign next to Connect to view its entries.

6 Select ContextLogoff.

7 Click Edit. The Edit Environment Profile window opens as shown in Figure 5.15.

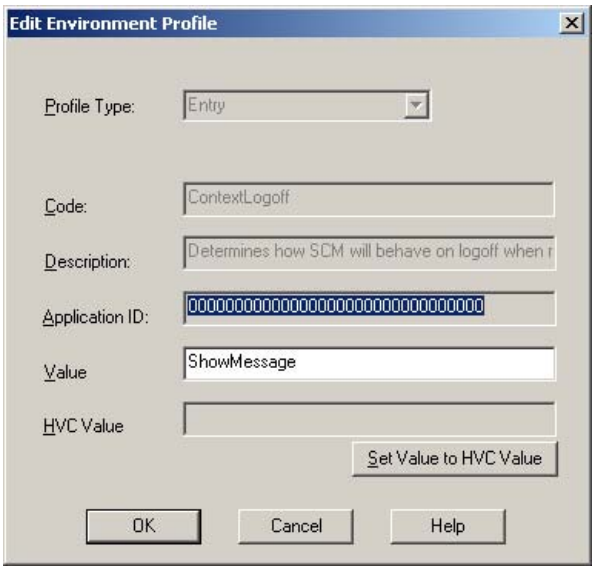


Figure 5.15: Edit Environment Profile Window

8 In the Value field, enter the one of the following options:

- ShowMessage: The following message appears in the window when SCM exits:

```
scm is currently in the clinical link.
Respond yes to logout all applications in the link.
Respond no to logout only scm.
```

- All: All participant applications exit.
- SCM: Only SCM exits, all other participant applications continue to run.

9 Click OK. Return to the Environment Profile Maintenance window.

10 Click Close.

This page contains Eclipsys proprietary information and is not to be duplicated or disclosed to unauthorized persons.

5.6.8 Configuring Sunrise Clinical Manager Context Controller

The purpose of the SCM Context Controller application is to enable the HL7 desktop integration functionality of SCM and to start it using automation. Although fully functional, it is essentially a sample of how the SCM context linking features are accessed.

The SCM Context Controller application is copied to the SCM program directory as part of the SCM Client installation. To use the SCM Context Controller application, simply configure it. If the SCM Context Controller is started without any command-line arguments, SCM is launched as a master participant with the Allow Data Loss option set to FALSE. For information on how SCM responds when a context change is requested, see "Responding to User Context Changes".

◆ To configure the SCM Context Controller

1 On the client workstation, navigate to the SCM program directory (this is typically C:\Program Files\Eclipsys Sunrise\Clinical Manager Client\version\).

2 Right-click the file scmcc.exe and select Create Shortcut. Put this shortcut on the desktop or into the Start menu so users can access it.

3 Right-click the shortcut file and select Properties from the shortcut menu.

4 Select the Shortcut tab. In the Target field, enter the command line parameter that indicates how you want SCM to participate in context sharing (see the table below). For example:

```
C:\Program Files\Eclipsys Sunrise\Clinical Manager Client\
version\scmcc.exe" /m /a-
```

Table 5.3 lists the Context Controller Command-Line Parameters.

Table 5.3: Parameter Listing

Parameter	Description
/m or /master	SCM is started as a master participant. This is the default if arguments are not specified.
/p or /peer	SCM is started as a peer participant.
/a or /a+ or /AllowDataLoss+	Sets the Allow Data Loss option to TRUE, which means that if there is unsaved data when the user selects to change the context from another application, that data is lost when the context changes or when logging off.
/a- or /AllowDataLoss- or <nothing>	Sets the Allow Data Loss option to FALSE. This is the default if arguments are not entered.

Note: If more than one type of participant type (master and peer) is specified, the last one is used.

This page contains Eclipsys proprietary information and is not to be duplicated or disclosed to unauthorized persons.

If more than one AllowDataLoss command (for example, /a and /a-) is specified, the last one is used.

If an unrecognized argument is entered, it is ignored.

By default, when the SCM Context Controller is started, it is not visible.

5.7 Managing the Clinical Link

When SCM is participating in context, the user has certain features available to them that allow them to use SCM in this mode.

5.7.1 Joining the Context

SCM attempts to join the context when it is launched using automation for the purpose of participating in the context. If joining the context is blocked for some reason, CPIO displays a message stating that context cannot be joined.

After the link is established, the context status indicators described below are set. If the context cannot be joined, the context status indicators are set accordingly (the Link-is-Broken icon) and the user is positioned in the Patient List chart section with an empty Patient List named "Clinical Link (Temporary)."

5.7.2 Viewing the Clinical Link State in Sunrise Clinical Manager

The clinical link can be in one of three states: on, broken, or changing. In SCM, the status of the clinical link appears by an icon on the Patient Header, below the patient's name, and as text in the status bar at the bottom of the window. These appear only when starting SCM by automation for the purpose of participating in context.



Figure 5.16: Sunrise Clinical Manager Window

5.7.3 Clinical Link Icons

When SCM is participating in the current context, the “Clinical Link is on” icon appears in the patient header, just below the patient's name. When SCM is not participating in the current context, the “Clinical Link is broken” icon appears.

This page contains Eclipsys proprietary information and is not to be duplicated or disclosed to unauthorized persons.

Table 5.4 shows the icons displayed in the Patient Header to indicate the state. These icons are standard and are specified by the HL7 committee.

Table 5.4: Link Icons

Icon	Meaning
	Clinical Link is on.
	Clinical Link is broken.
	Clinical Link is changing.

5.7.4 The Clinical Link Menu

When SCM has been started by automation for the purpose of participating in context, the Clinical Link menu appears in the menu bar.

Table 5.5: Clinical Link Details

Menu	Command
Clinical Link	Rejoin Clinical Link
	Use This_Application's Data: Rejoins and resets the context to that of SCM. The italicized text is replaced with the patient's name and, if appropriate, the visit ID. The visit ID is included if both contexts (the application's and the global) are for the same patient ID.
	Use Common Context Data: Rejoins using the current global settings. If this application's data is used, the context change is published.
	If the global context is not valid, the global option is disabled.
	Break Clinical Link: Removes SCM as a participant in the context.

5.7.5 Rejoining the Context

When the clinical link is broken, request that the link be reestablished.

◆ To rejoin the context

From the Clinical Link menu, select Rejoin Clinical Link and one of the following options:

This page contains Eclipsys proprietary information and is not to be duplicated or disclosed to unauthorized persons.

- Select Use This_Application's Data to rejoin and reset the context to that of SCM. The text in the brackets is replaced with the patient's name and, if appropriate, the visit ID from the context.
- Select Use Common Context Data to rejoin using the current global settings. If this application's data is used, the context change is published.

5.7.6 Setting the View

The view context item allows an application to specify where the user should be positioned when the application is opened. For example, in the case of SCM, an application might want the user positioned in the SCM Orders chart section when the application is opened. Setting the view allows an application to determine which application should become active and where in that application the user should be positioned when the context is changed. This allows one application to open another application. The other application must already be started on the desktop.

SCM supports setting the view to any of the chart sections based on the chart section caption. This includes setting the view to a custom chart section. The default view is the Patient List. If there is a problem with setting the view to another chart section, the view is not changed.

5.7.7 Patient List Features for Clinical Link List

The Clinical Link (Temporary) Patient List is patterned on the existing temporary Patient List. It is a hard-coded list, not a personal preference. It only exists in the user's drop down list when SCM has been started for the purpose of participating in context. Unlike a temporary list, it always contains visits for a single patient (the one currently in the global context). If the patient does not exist or the user does not have access to any of the visits, the message "No patient visits found for this list" appears.

5.8 Responding to Context Changes

5.8.1 Responding to User Context Changes

SCM responds to User Context Changes only when running in peer-to-peer mode. When the user context changes, SCM attempts to log the user in with the supplied user name and password (the old user is first logged out).

The messages in Table 5.6 appear when the peer application initiates user context changes.

This page contains Eclipsys proprietary information and is not to be duplicated or disclosed to unauthorized persons.

Table 5.6: Message List

Condition	Description
Unable to Log Out: Unsubmitted Data	If SCM receives a request for a user context change and there is unsubmitted data on one of the modeless windows (for example, the Order Entry Worksheet, Document Entry Worksheet, and so on) and the AllowDataLoss flag is set to FALSE, SCM sends back a warning that there is unsaved data. If the user continues with the user context change, SCM breaks the context link and allows the user to submit the data. If the AllowDataLoss Flag is set to TRUE, the user does not get the warning and any unsaved data is lost.
Unable to Log Out: Modal Dialog Present	If SCM receives a request for a user context change and a modal window is open, SCM sends back a warning that there is a problem with changing the context. If the user continues with the user context change, all modal windows are closed and any unsaved data in them is lost.
Unable to Log In	If the user does not exist or the password is not valid, SCM displays the logon window. The link is broken and SCM does not respond to context changes until a new user logs on. After a new user logs on to SCM, the new user is published into context.

5.8.2 Responding to Patient Context Changes

SCM responds to patient context changes only in peer-to-peer modes. It does not respond to patient context changes when running in master mode. When the patient context changes, SCM attempts to switch to the new patient. Table 5.7 describes how SCM handles the condition listed should it occur when the context change is changed by the peer application.

Table 5.7: Context Change Listing (1 of 2)

Condition	Description
Patient ID (MRN) does not Exist	In this scenario, because the currently selected patient in SCM does not have a MRN, the patient is not published to context.
Patient ID (MRN) does not Match Patient in Database	If the patient ID does not match a known patient, the user is positioned on the Clinical Link list with the standard message "No patient visits found for this list."

This page contains Eclipsys proprietary information and is not to be duplicated or disclosed to unauthorized persons.

Table 5.7: Context Change Listing (2 of 2)

Condition	Description
No Visit ID in Context	SCM is visit driven because a user first selects a visit to gain access to the patient data. If no visit ID is present and a valid patient ID exists, the application opens at the Patient List. A temporary list named Clinical Link (Temporary) is created populated with a list of all the known visits for the patient. SCM's security features are enforced. Specifically, a visit does not appear unless the user has the appropriate rights to access it.
User Does not Have Access to Patient or Visit	The user is positioned on the Clinical Link list with the standard message "No patient visits found for this list."
Cannot Change Context - Unsaved Data Present	SCM receives a context change request while there is unsaved data present. See the section below on "Changing the Context with Unsaved Data" for details on the behavior of SCM.

5.8.3 Changing the Context with Unsaved Data

5.8.3.1 Unsaved Data on the Manager Windows

If the user decides to change the context even though SCM has unsaved data on the various Manager windows (for instance, Order Entry Worksheet, Worklist Manager, and so on), SCM behaves differently depending on the value of the AllowDataLoss parameter.

If the AllowDataLoss flag is FALSE, and the user responds OK to the message: “Unsubmitted orders will be lost, review order worksheet”, SCM minimizes itself. The link is broken and the Clinical Link is Broken window appears informing the user why. When the user restores SCM, they must save or discard the data, and then use the Rejoin Clinical Link menu option to rejoin the clinical link.

If the AllowDataLoss flag is TRUE, then SCM deletes any unsaved data and context changes in SCM.

The message can appear in Manager windows (for instance, Order Entry Worksheet, Worklist Manager, and so on). An example is shown in Figure 5.17.

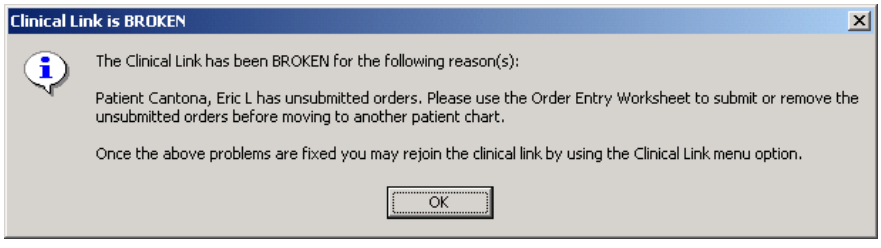


Figure 5.17: Clinical Link is Broken Message

This page contains Eclipsys proprietary information and is not to be duplicated or disclosed to unauthorized persons.

5.8.3.2 Cannot Change Context: Modal Window is Open

SCM displays a message, stating that a window is open and data can be lost. If the user proceeds, the window is closed automatically. If the window had any unsaved data, that data is lost.

Windows that fall in this category include, but are not limited to, error messages and patient information data entry windows. The Manager windows (Order Entry Worksheet, Worklist Manager, and so on) are modeless and do not fall into this category.

5.8.3.3 Messages Supplied by Sunrise Clinical Manager

Example messages appear in SCM when there is a problem with changing the user or patient context:

- Unsubmitted Orders: Unsubmitted orders are lost. Review order worksheet.
- Unsubmitted Documents: Unsubmitted documents are lost. Review doc worksheet.
- Modal window open: A window is open that can have unsaved data. Close the window or data can be lost.
- Multiple Problems: Unsubmitted data can be lost. Save the data before proceeding.

Note: These messages are examples only and do not represent a complete list.

5.8.3.4 Other Application Identifies Problem with Context Change Request

If SCM attempts to change the context, other applications are polled and can reply that the user could have a problem doing so, or are busy and not processing messages. The HL7 standard specifies that the user be presented with a window and given the choice about what to do.

The Problem Changing Clinical Data window below shows an example of the message SCM displays when at least one participating application reports a problem in changing the context, and none of the applications is busy. The text following the application name is supplied by the application.

This page contains Eclipsys proprietary information and is not to be duplicated or disclosed to unauthorized persons.

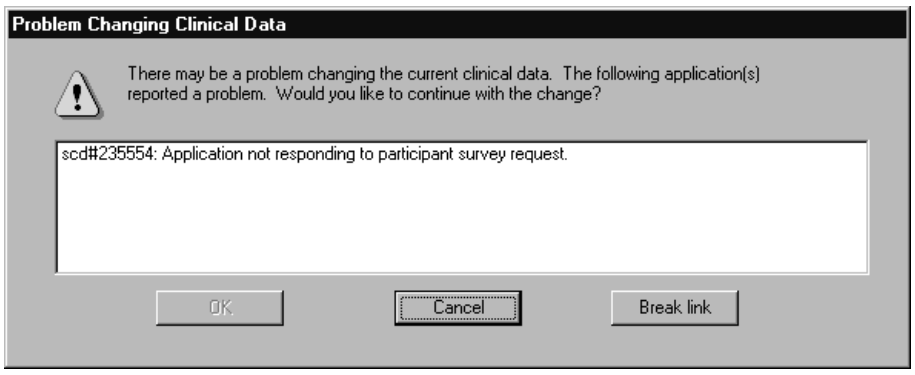


Figure 5.18: Problem Changing Clinical Data Window

Table 5.8: Window Elements

Feature	Description
Message	Describes the problem.
Application Names	Supplied by the Context Manager.
Application Message	A message from the application stating the problem.
OK	Allows the context to change. The window closes and the patient is changed as requested. **Note:** This option is disabled when at least one application is busy. Cancel the request or break the link.
Cancel	Cancels the request to change the context. The window closes and the patient is unchanged.
Break Link	SCM breaks the link.

5.8.4 Changing the Context When Sunrise Clinical Manager is Minimized

SCM applies all context changes even if it has been minimized. If the minimized application has broken the link, a message appears describing the nature of the problem. After the problem has been corrected, the user is given the opportunity to rejoin the context.

5.8.5 Changing the Context When Sunrise Clinical Manager is Suspended

If SCM is Suspended, either manually by the user or automatically because of a timeout, it is removed as a context participant and no further context changes are sent to it. When the user Unsuspends the session, SCM rejoins context as a participant and publish its current context.

This page contains Eclipsys proprietary information and is not to be duplicated or disclosed to unauthorized persons.

Only one instance of SCM can participate in context at a time. In peer mode, if a second instance of SCM is started while another instance is running, the first instance are suspended and it is removed as a participant so no further context changes are sent to it.

The second instance of SCM then joins context as a participant and apply and publish context changes. When the first instance is unsuspended, the second instance is then suspended. The first instance then publishes its user and patient to context.

This page contains Eclipsys proprietary information and is not to be duplicated or disclosed to unauthorized persons.

CHAPTER 6: CREATING AN ACTIVEX CUSTOM TAB

Contents

6.1 Overview 105
6.2 The Shell Data Sink Interfaces 106
6.3 The ActiveX Control Interfaces 110
6.4 Using C++ 121
6.5 Getting Started With Visual Basic 122
6.6 Accessing Sunrise Clinical Manager Data 124
6.7 Controlling Tabbing Behavior 126
6.8 Obtaining Data From the Sunrise Clinical Manager Database 128
6.9 Adding New ActiveX Tab Entries to Sunrise Clinical Manager Environment Profiles 137

6.1 Overview

A user can add one or more custom tabs known to the Sunrise Clinical Manager user as chart sections to the standard SCM user interface. Each of these tabs can host a single ActiveX control and behave in a manner similar to the standard tabs.

Functions are provided to retrieve context data such as the current patient, current user, current database, and server. Other functions populate the Actions menu and notify the tab that the user has changed the current patient. These functions are accessible using C++ or Visual Basic and are described in the rest of this chapter.

Figure 6.1 shows a high-level view of the interactions of the custom tab with the rest of the SCM application.

This page contains Eclipsys proprietary information and is not to be duplicated or disclosed to unauthorized persons.

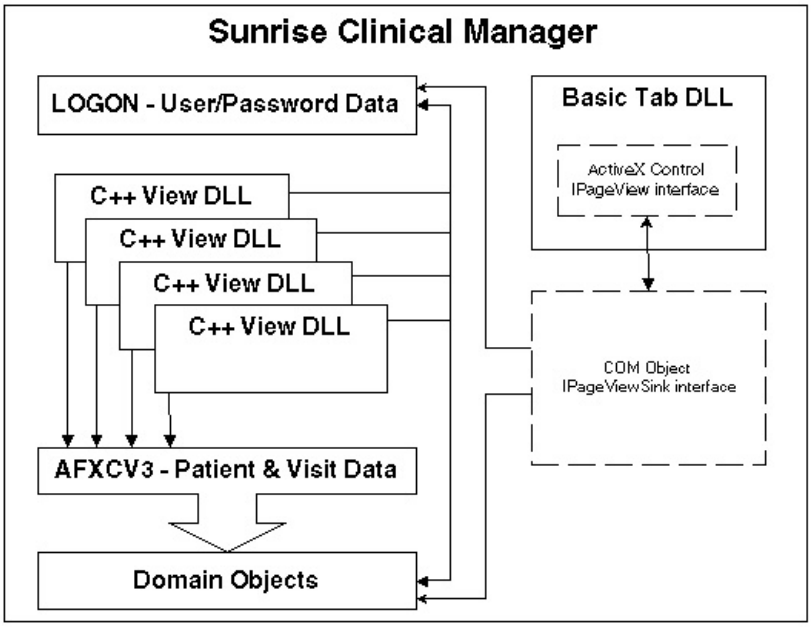


Figure 6.1: High Level Interactions

CBasicTabView uses the virtual methods of its parent CHVCPageView class by delegating implementation to the IPageView interface used in the ActiveX control. When the ActiveX control is created, the tab view requests the sink from CCV3MainFrame in the SCM shell.

The sink is connected to the ActiveX control. When the patient and visit data are updated by user actions in the shell, the ActiveX control is notified through the OnChangeVisit method of the IPageView interface. The ActiveX control retrieves the data from the shell through the IPageViewSink interface.

After creating the Active X control application, use SCM Configuration Tools to add the entry to the environment profile.

6.2 The Shell Data Sink Interfaces

The COM object CoPageViewSink in hvc.exe implements these two interfaces internally:

- IPageViewSinkSet is an interface used privately by C++ code in hvc.exe to populate connection, patient, and visit data.
- IPageViewSink provides a read-only interface for retrieving the data in CoPageViewSink. This interface is compliant with COM programming environments that are restricted to automation data types.

```
interface IPageViewSinkSet : IUnknown
{
HRESULT SetValues(LPOLESTR* pwcsNames, LPOLESTR* pwcsValues,
long nValues);
```


This page contains Eclipsys proprietary information and is not to be duplicated or disclosed to unauthorized persons.

```
}

interface IPageViewSink : IUnknown
{
HRESULT GetPatientRecordNames([out,retval]SAFEARRAY(BSTR)*
Names);
HRESULT GetPatientRecordValues([in]SAFEARRAY(BSTR)* Names,
[out,retval]SAFEARRAY(BSTR)* Values);
};
```

Figure 6.2: Shell Data Sink Interfaces

6.2.1 IPageViewSink::GetPatientRecordNames

GetPatientRecordNames returns an array of strings, which contain all the names with matching values in the SCM.

6.2.1.1 Visual C++ Syntax

```
HRESULT GetPatientRecordNames(
[out,retval]SAFEARRAY(BSTR)* Names);
```

Where:

Names = [out]

A string array containing all the names for associated values which can be requested from the SCM shell.

RETURN VALUES

GetPatientRecordNames has three return values.

Table 6.1: Return Values for GetPatientRecordNames

HRResult	Description
S_OK	The method succeeded and has returned the expected value or values.
E_OUTOFMEMORY	Memory allocation error occurred when allocating array.
E_FAIL	Error creating the array.

REMARKS

GetPatientRecordNames provides the ActiveX tab with an array containing the names associated with values in the SCM current context. Using the name, the ActiveX tab retrieves the associated value using the GetPatientRecordValues method on the IPageViewSink interface.

This page contains Eclipsys proprietary information and is not to be duplicated or disclosed to unauthorized persons.

This interface is passed to the ActiveX control through IPageView::AdviseSink. Table 6.2 lists the available attributes in the current context.

Table 6.2: Attribute Table (1 of 2)

Attribute	Description
DBDRIVER	The name of the database driver. This sometimes is needed to establish a connection to the database.
DBSERVER	The name of the database server to which this session of SCM is connected. This sometimes is needed to establish a connection to the database.
DBNAME	The name of the database to which this session of the SCM is connected. This sometimes is needed to establish a connection to the database.
PATIENTID	The patient MRN number for the current visit as shown in the Patient Header.
VISITNUMBER	The visit number for the current visit as shown in the Patient Header.
PATIENTGIVENNAME	The patient's given name as shown in the Patient Header.
PATIENTMIDDLENAME	The patient's middle name as shown in the Patient Header
PATIENTLASTNAME	The patient's last name as shown in the Patient Header
PATIENTLOCATIONFACILITYID	The facility of the current visit's assigned location.
PATIENTLOCATIONUNIT	The unit of the current visit's assigned location.
PATIENTLOCATIONROOM	The room of the current visit's assigned location.
PATIENTLOCATIONBED	The bed of the current visit's assigned location.
PROVIDERID-xxx	A list of the care provider IDs for the current user, where xxx is the care provider ID type. For example, PROVIDERID-PRIMARY would be the care provider ID of the type Primary. If the user does not have any care provider IDs, this entry does not exist.

This page contains Eclipsys proprietary information and is not to be duplicated or disclosed to unauthorized persons.

Table 6.2: Attribute Table (2 of 2)

Attribute	Description
SESSIONID	The ID of the current context session. If multiple sessions of SCM are running on the desktop, each has its own session ID. This value is useful if connecting to the context outside of a custom tab.
USERID	The ID of the current user. This is the value the user types in at logon.
USERNAME	The user's full name in the format Last, Given Middle.

SAMPLE VISUAL BASIC CODE

For examples of how this function is used, see "Accessing Sunrise Clinical Manager Data" on page 124.

SEE ALSO

GetPatientRecordNames, IPageView::AdviseSink, IPageView::UnadviseSink

6.2.2 IPageViewSink::GetPatientRecordValues

GetPatientRecordValues returns an array of values matching the names provided in the name array.

6.2.2.1 Visual C++ Syntax

```
HRESULT GetPatientRecordValues(
[in]SAFEARRAY(BSTR)* Names,
[out,retval]SAFEARRAY(BSTR)* Values);
```

Where:

Names = [in]

A string array containing only names for associated values which can be requested from the SCM shell.

Values = [out]

A string array containing the associated values matching the name in the Names array.

This page contains Eclipsys proprietary information and is not to be duplicated or disclosed to unauthorized persons.

Return Values

GetPatientRecordValues has three return values.

Table 6.3: Return Values for GetPatientRecordValues

HRResult	Description
S_OK	The method succeeded and has returned the expected value or values.
E_OUTOFMEMORY	Memory allocation error occurred when allocating array.
E_FAIL	Error creating the array.

Remarks

GetPatientRecordValues provides the ActiveX tab with an array containing the patient record values available from SCM. To determine the names for this method, use IPageViewSink::GetPatientRecordNames. This interface is passed to the ActiveX control through IPageView::AdviseSink.

Sample Visual Basic Code

For examples of how this function is used, see "Accessing Sunrise Clinical Manager Data" on page 124.

See Also

GetPatientRecordNames, IPageView::AdviseSink, IPageView::UnadviseSink

6.3 The ActiveX Control Interfaces

The ActiveX control participates in tab events by implementing the IPageView interface.

```
interface IPageView : IUnknown
{
// Base virtual functions
HRESULT GetTabCaption([out]BSTR* bstrReport);
HRESULT GetActionMenu([out]SAFEARRAY*(BSTR));
HRESULT FireActionMenuItem([in]long nID);

// Data Integrity
HRESULT CanChangePatient();
HRESULT CanCloseTab();

// Navigation methods
HRESULT SetFocusFirst();
HRESULT SetFocusLast();
HRESULT IsFirstWnd();
HRESULT IsLastWnd();

// Shell Info
 HRESULT AdviseSink([in]IPageViewSink* PageViewSink,
```


This page contains Eclipsys proprietary information and is not to be duplicated or disclosed to unauthorized persons.

```
 [out,retval]long* ConnectId);
 HRESULT UnadviseSink([in]long ConnectId);
 HRESULT OnChangeVisit();
};
```

Figure 6.3: ActiveX Control Interface

6.3.1 IPageView::AdviseSink

AdviseSink connects the sink interface in the SCM shell to the ActiveX tab.

6.3.1.1 Visual C++ Syntax

```
HRESULT AdviseSink([in]IPageViewSink* PageViewSink,
[out,retval]long* ConnectId);
```

Where:

PageViewSink = [in]

The interface pointer from the SCM shell, which provides the ActiveX control with the ability to retrieve patient record data from the shell.

ConnectId = [out]

The unique identifier for the connection, which is used to release the SCM sink interface.

RETURN VALUES

AdviseSink has two return values.

Table 6.4: Return Values for AdviseSink

HRResult	Description
S_OK	The method succeeded and has returned the expected value or values.
E_FAIL	The ActiveX tab already has a connection.

REMARKS

The SCM shell uses AdviseSink to connect the patient record sink to the ActiveX tab. The ActiveX tab can then call methods on the IPageViewSink interface to retrieve data from the available data elements and configuration information from the SCM shell.

This function is used to create a connection from the ActiveX control to the SCM application. After the connection is established, it is used to obtain data elements made available by SCM.

Create this connection to SCM by creating a variable of type IpageViewSink, then setting the value of that variable to the PageViewSink parameter of the AdviseSink function. The list of available data element names or attributes is fixed and can be obtained using the function GetPatientRecordNames. The values for

This page contains Eclipsys proprietary information and is not to be duplicated or disclosed to unauthorized persons.

each of those data elements can be obtained using the GetPatientRecordValues function. For more information, see "Accessing Sunrise Clinical Manager Data" on page 124.

SAMPLE VISUAL BASIC CODE

```
Implements IPageView
Dim mySink as IPageViewSink

Private Function IPageView_AdviseSink(ByVal PageViewSink As _
 BASICTABLib.IPageViewSink) As Long
Set mySink = PageViewSink
End Function
```

Figure 6.4: AdviseSink Visual Basic Code Example

6.3.2 IPageView::UnadviseSink

UnadviseSink provides the SCM shell the ability to disconnect its sink interface from the ActiveX tab.

6.3.2.1 Visual C++ Syntax

```
HRESULT UnadviseSink([in]long ConnectId);
```

Where:

ConnectId = [in]

The connection identifier provided by an earlier call to AdviseSink.

RETURN VALUES

UnadviseSink has two return values.

Table 6.5: Return Values for UnadviseSink

HRResult	Description
S_OK	The method succeeded and has returned the expected value or values.
E_FAIL	The connection identifier does not identify an existing connection.

REMARKS

The SCM shell calls AdviseSink to connect the patient record sink to the ActiveX tab. This returns a connection identifier. When the SCM shell wants to disconnect the sink from the ActiveX tab, it calls UnadviseSink with this identifier.

This function is used by SCM to advise the ActiveX control that the connection is being broken. This occurs when SCM is shutting down. Use this function to release the object variable that was allocated for the connection.

SAMPLE VISUAL BASIC CODE

```
Private Sub IPageView_UnadviseSink(ByVal ConnectId As Long)
```


This page contains Eclipsys proprietary information and is not to be duplicated or disclosed to unauthorized persons.

```
Set mySink = Nothing
End Sub
```

Figure 6.5: UnadviseSink Visual Basic Code Example

SEE ALSO

AdviseSink, IPageViewSink::GetPatientRecordNames, IPageViewSink::GetPatientRecordValues

6.3.3 IPageView::OnChangeVisit

The SCM shell calls OnChangeVisit to notify the ActiveX control that the currently selected patient visit has changed. This event is also generated when logging on, when the user changes, and when the user moves from tab to tab.

6.3.3.1 Visual C++ Syntax

```
HRESULT OnChangeVisit();
```

RETURN VALUES

OnChangeVisit has one return value.

Table 6.6: Return Values for OnChangeVisit

HRResult	Description
S_OK	The method succeeded and has returned the expected value or values.

REMARKS

Unlike CanChangePatient, which allows the ActiveX tab to advise the SCM shell that it has work to complete, this method simply informs the ActiveX tab that the patient, user, or tab has changed so it can do work like clear caches or retrieve new data.

Because this event is triggered every time the user changes tabs, check to see that the patient or visit has actually changed before processing the event.

SAMPLE VISUAL BASIC CODE

```
Private Sub IPageView_OnChangeVisit()
...check to see if the visit has really changed by comparing
the previous values for patient ID, visit ID and user ID to
see if the are different from the current values and that
they are not blank.

If bNewVisit then
...do something to update the data in the control to the new
patient's data if the visit has changed
End If

End Sub
```

Figure 6.6: OnChangeVisit Visual Basic Code Example

This page contains Eclipsys proprietary information and is not to be duplicated or disclosed to unauthorized persons.

SEE ALSO

CanChangePatient

6.3.4 IPageView::GetActionMenu

GetActionMenu provides an array of strings that the ActiveX tab housing converts into an Actions submenu.

6.3.4.1 Visual C++ Syntax

```
HRESULT GetActionMenu(
SAFEARRAY*(BSTR) psaMenuItems);
```

Where:

psaMenuItems = [out]

The safearray contains an array of strings. These strings are used to populate a submenu placed under the Actions menu item in the SCM shell. Only a single dimension array is provided.

RETURN VALUES

GetActionMenu has two return values.

Table 6.7: Return Values for GetActionMenu

HRResult	Description
S_OK	Safearray is allocated with MenuItem strings.
S_FALSE	No safearray is allocated. There is no menu for this tab.

REMARKS

If the ActiveX control is to provide and respond to menu selections from users, it must provide strings for a menu. The SCM constructs a menu from the array of strings and associates them with MenuItemIDs. These items then appear in the Actions menu of the SCM shell when the tab that contains the ActiveX control is the active tab.

The zero-based subscript for the string is returned in a call to FireActionMenuItem if the user selects an item from the Actions menu.

To add a separator bar, give the menu item the name "SEPARATOR" using all caps. If a menu accelerator key is needed, put an ampersand (&) in front of the letter to use.

No provision has been made to support submenus.

This page contains Eclipsys proprietary information and is not to be duplicated or disclosed to unauthorized persons.

SAMPLE VISUAL BASIC CODE

The following code adds four menu items to the Actions menu in SCM. The third and fourth items have a separator line between them. The ampersand (&) preceding the number in the menu name causes the number to be underlined in the menu. Users can then use Alt plus the underlined letter to access that item when the menu is open.

```
Private Function IPageView_GetActionMenu() As String()
Dim sActionMenu(4) As String
sActionMenu(0) = "Sample Menu &0"
sActionMenu(1) = "Sample Menu &1"
sActionMenu(2) = "Sample Menu &2"
sActionMenu(3) = "SEPARATOR"
sActionMenu(4) = "Sample Menu &4"

IPageView_GetActionMenu = sActionMenu
End Function
```

Figure 6.7: GetActionMenu Visual Basic Code Example

SEE ALSO

FireActionMenuItem

6.3.5 IPageView::FireActionMenuItem

FireActionMenuItem is called by the SCM shell when a user selects an Actions menu item provided by the ActiveX control.

6.3.5.1 Visual C++ Syntax

```
HRESULT FireActionMenuItem(
long nID);
```

Where:

nID = [in] Subscript identifying the menu selection made by the user.

RETURN VALUES

FireActionMenuItem has three return values.

Table 6.8: Return Values for FireActionMenuItem

HRResult	Description
S_OK	Menu selection was properly fired.
E_INVALID	nID is an invalid subscript for the menu.
E_UNEXPECTED	Menu has not been initialized by a call to GetActionMenu.

This page contains Eclipsys proprietary information and is not to be duplicated or disclosed to unauthorized persons.

REMARKS

FireActionMenuItem returns the value, which is the zero-based subscript of the string provided for the Actions submenu in the call to GetActionMenu. For example, the safearray of strings contains &List, &Search, and &Find. These items are associated with the values of 0, 1, and 2 respectively.

SAMPLE VISUAL BASIC CODE

The following code shows obtains the MenuID of the Actions menu selection that a user has made. Any separator bars are counted in the sequence of menu options, even though the user cannot select them.

The MenuID numbers correspond to the array index value of the menu array that created the menu. See the Visual Basic example in GetActionMenu.

```
Private Sub IPageView_FireActionMenuItem(ByVal MenuID As Long)

Select Case MenuID
 Case 0
 ...do something for the first menu option
 Case 1
 ...do something for the second menu option
 Case ...
 ...do something for each of the n menu options
 Case Else
 ...do something else in case you missed an option
End Select
End Sub
```

Figure 6.8: FireActionMenuItem Visual Basic Code Example

SEE ALSO

GetActionMenu

6.3.6 IPageView::CanChangePatient

CanChangePatient is called when the user attempts to change the patient but before the patient is actually changed.

6.3.6.1 Visual C++ Syntax

```
HRESULT CanChangePatient();
```

RETURN VALUES

CanChangePatient has two return values.

Table 6.9: Return Values for CanChangePatient

HRResult	Description
S_OK	The patient change can occur.

This page contains Eclipsys proprietary information and is not to be duplicated or disclosed to unauthorized persons.

Table 6.9: Return Values for CanChangePatient

HRResult	Description
S_FALSE	The tab still has work to be completed.

REMARKS

CanChangePatient allows the tab to prompt the user to save any outstanding work before changing to a new patient/visit. If the tab can commit work without user intervention, it should do so without notifying the user and returns S_OK (this is the default if no explicit response is given).

In Visual Basic, the ActiveX control responds to the SCM request to change the patient by passing back an error object if the change should not be allowed. The error object has three parameters:

- Error number a number between 513 and 65535 which is added to the standard variable vbObjectError. It has no meaning, and does not appear anywhere, but it must be present.
- Source of the error appearing in the title bar of the error message box.
- Description of the error appearing in the body of the error message box.

SAMPLE VISUAL BASIC CODE

```
Dim bAllowPatientChange as Boolean
...do something here to set the value of bAllowPatientChange
Private Sub IPageView_CanChangePatient()
Dim iErrorNumber as integer
Dim sSource as string
Dim sMessage as string

If bAllowPatientChange = False Then
 IErrorNumber = 600'This must be a number between 513 and
                  '65536
 sSource = "MyTab"
 SMessage = "You have unsaved data on MyTab"

 Err.Raise vbObjectError + iErrorNumber, sSource , sMessage

End If

End Sub
```

Figure 6.9: CanChangePatient Visual Basic Code Example

SEE ALSO

CanCloseTab

This page contains Eclipsys proprietary information and is not to be duplicated or disclosed to unauthorized persons.

6.3.7 IPageView::CanCloseTab

CanCloseTab is called prior to the SCM shell closing. This is typically a result of a log off request by the user.

6.3.7.1 Visual C++ Syntax

```
HRESULT CanCloseTab();
```

RETURN VALUES

CanCloseTab has two return values.

Table 6.10: Return Values for CanCloseTab

HRResult	Description
S_OK	The tab can be closed.
S_FALSE	The tab has work that is not yet completed.

REMARKS

CanCloseTab allows the tab to prompt the user to save any outstanding work done on the clinical record before closing the SCM shell. If the tab can commit work without user intervention, it should do so without notifying the user and then return S_OK.

SAMPLE VISUAL BASIC CODE

In Visual Basic, the ActiveX control responds to the SCM request to close the application by passing back an error object if the change should not be allowed. The error object has three parameters:

- Error number: a number between 513 and 65535 which is added to the standard variable vbObjectError. It has no meaning, and does not appear anywhere, but it must be present.
- Source of error: appearing in the title of the error message box.
- Description of error: appearing in the body of the error message box.

```
Dim bAllowCloseTab as Boolean

...do something here to set the value of bAllowCloseTab

Private Sub IPageView_CanCloseTab()
Dim iErrorNumber as integer
Dim sSource as string
Dim sMessage as string
If bAllowCloseTab = False Then
 IErrorNumber = 600'This must be a number between 513 and'65536
 sSource = "MyTab"
 SMessage = "You have unsaved data on MyTab"

 Err.Raise vbObjectError + iErrorNumber, sSource, sMessage
```


This page contains Eclipsys proprietary information and is not to be duplicated or disclosed to unauthorized persons.

```
End If

End Sub
```

Figure 6.10: CanCloseTab Visual Basic Code Example

SEE ALSO

CanChangePatient

6.3.8 IPageView::SetFocusFirst

SetFocusFirst sets the window focus on the first user entry window in the ActiveX control.

6.3.8.1 Visual C++ Syntax

```
HRESULT SetFocusFirst();
```

RETURN VALUES

SetFocusFirst has one return value.

Table 6.11: Return Values for SetFocusFirst

HRResult	Description
S_OK	Focus set on first window in ActiveX control tab.

REMARKS

The ActiveX control must always return S_OK from this result. If an ActiveX control hosted in a tab does not provide any windows for user entry, the control need not support the IPageView interface. The SCM shell calls this function when a user presses the Tab key when the focus is on the tab control itself.

SAMPLE VISUAL BASIC CODE

See "Controlling Tabbing Behavior" on page 126 for examples of how to use this function.

6.3.9 IPageView::SetFocusLast

SetFocusLast sets the window focus on the last user entry window in the ActiveX control.

6.3.9.1 Visual C++ Syntax

```
HRESULT SetFocusLast();
```


This page contains Eclipsys proprietary information and is not to be duplicated or disclosed to unauthorized persons.

RETURN VALUES

SetFocusLast has one return value.

Table 6.12: Return Values for SetFocusLast

HRResult	Description
S_OK	Focus set on last window in ActiveX control tab.

REMARKS

The ActiveX control must always return S_OK from this result. If an ActiveX control hosted in a tab does not provide any windows for user entry, the control need not support the IPageView interface. When the focus is on the tab control itself, the SCM shell calls this function when a user presses Shift+Tab.

SAMPLE VISUAL BASIC CODE

See "Controlling Tabbing Behavior" on page 126 for information on how to use this function.

6.3.10 IPageView::IsFirstWnd

IsFirstWnd determines whether input focus is in the first entry window in the ActiveX control.

6.3.10.1 Visual C++ Syntax

```
HRESULT IsFirstWnd();
```

RETURN VALUES

IsFirstWnd has two return values.

Table 6.13: Return Values for IsFirstWnd

HRResult	Description
S_OK	The focus is currently in the first user entry window in the ActiveX control.
S_FALSE	The focus is not currently in the first user entry window in the ActiveX control.

REMARKS

The SCM control calls this function when the user presses Shift+Tab. If the call returns S_OK, the SCM shell changes the focus to the tab control in the shell from the ActiveX control. Use this method to ensure that the ActiveX control and the tab control in the shell participate in proper tab ordering.

This page contains Eclipsys proprietary information and is not to be duplicated or disclosed to unauthorized persons.

SAMPLE VISUAL BASIC CODE

See "Controlling Tabbing Behavior" on page 126 for information on how to use this function.

6.3.11 IPageView::IsLastWnd

IsLastWnd determines whether input focus is in the last entry window in the ActiveX control.

6.3.11.1 Visual C++ Syntax

```
HRESULT IsLastWnd();
```

RETURN VALUES

IsLastWnd has two return values.

Table 6.14: Return Values for IsLastWnd

HRResult	Description
S_OK	The focus is currently in the last user entry window in the ActiveX control.
S_FALSE	The focus is not currently in the last user entry window in the ActiveX control.

REMARKS

The SCM control calls this function when the user presses Tab. If the call returns S_OK, the SCM shell changes the focus to the tab control in the shell from the ActiveX control. Use this method to ensure that the ActiveX control and the tab control in the shell participate in proper tab ordering.

SAMPLE VISUAL BASIC CODE

See "Controlling Tabbing Behavior" on page 126 for information on how to use this function.

6.4 Using C++

Each custom tab added to the SCM shell must be associated with exactly one ActiveX control. The only requirements on this control are that it implements the custom IPageView COM interface, and that the control be registered on the workstation. Eclipsys provides the definition of this interface using the type library embedded inside BasicTab.dll.

This page contains Eclipsys proprietary information and is not to be duplicated or disclosed to unauthorized persons.

Visual C++ developers can generate the necessary header files in either of two ways:

- Using the #import directive. The statement #import "BasicTab.dll"

  generates two files, BasicTab.tlh and BasicTab.tli, which are used for implementing the IPageView interface, and for calling methods on the outgoing IPageViewSink interface.

- Using the OLE/COM Object Viewer utility shipped with Visual C++, to extract the equivalent IDL from the type library. This IDL file can then be compiled using MIDL to generate the necessary files. For example:

```
midl.exe basictab.idl /iid ..\include\basictab_i.c /h
..\include\basictab.h     /tlb .\basictab.tlb
```

Figure 6.11: MIDL File Generation

The /iid and /h compiler switches specify the filenames for the generated files. For further information, consult the official Microsoft documentation on #import and MIDL.

6.5 Getting Started With Visual Basic

To access the functions of the custom tab from Visual Basic, use Visual Basic 6.0 or higher.

◆ To get started with Visual Basic:

1 Create a new project of the User Control type.

2 From the Project menu, select References and add BasicTab.dll as the reference to the SCM Basic Tab Library. Browse to the SCM client directory to locate the BasicTab.dll file.

3 Add the following line to the declaration section of the project:

```
Implements IpageView
```

4 In the editor, create and use any of the functions (see "Available Functions" in this section):

a From the Object drop-down list, select IpageView.

b In the Procedure list, select, in turn, each of the appropriate procedures. Details of each function are described in "The Shell Data Sink Interfaces" on page 106 and "The ActiveX Control Interfaces" on page 110.

This page contains Eclipsys proprietary information and is not to be duplicated or disclosed to unauthorized persons.

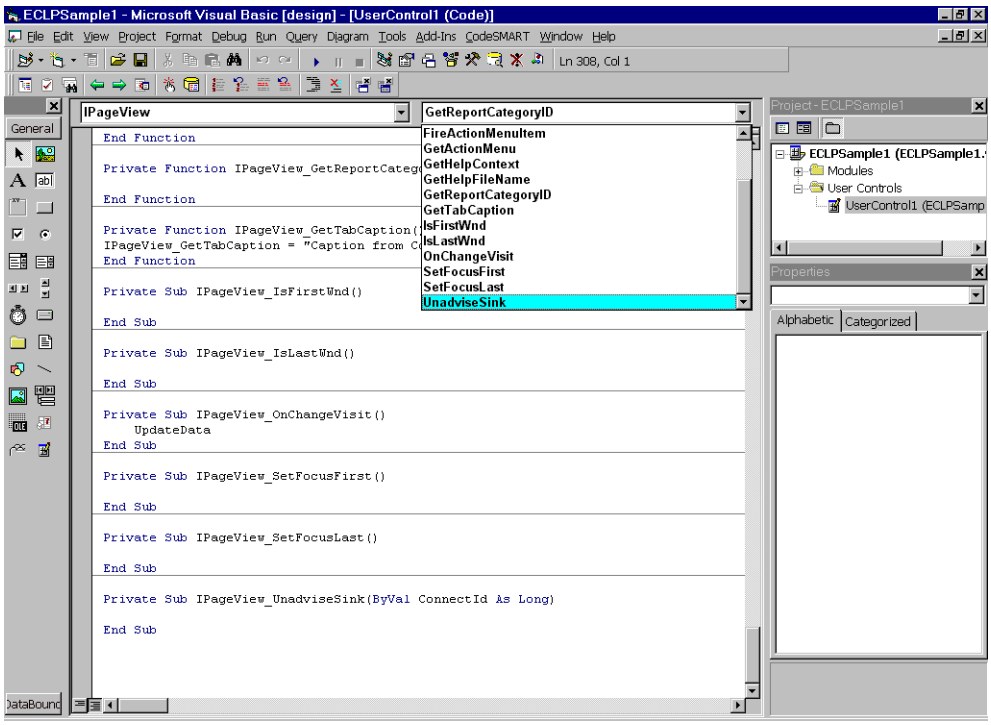


Figure 6.12: Visual Basic Example

AVAILABLE FUNCTIONS

These functions are available with Visual Basic.

```
Private Function IPageView_AdviseSink(ByVal PageViewSink As
BASICTABLib.IPageViewSink) As Long

End Function

Private Sub IPageView_CanChangePatient()

End Sub

Private Sub IPageView_CanCloseTab()

End Sub

Private Sub IPageView_FireActionMenuItem(ByVal MenuID As Long)

End Sub

Private Function IPageView_GetActionMenu() As String()

End Function

Private Function IPageView_GetTabCaption() As String

End Function
```


This page contains Eclipsys proprietary information and is not to be duplicated or disclosed to unauthorized persons.

```
Private Sub IPageView_IsFirstWnd()

End Sub

Private Sub IPageView_IsLastWnd()

End Sub

Private Sub IPageView_OnChangeVisit()

End Sub

Private Sub IPageView_SetFocusFirst()

End Sub

Private Sub IPageView_SetFocusLast()

End Sub

Private Sub IPageView_UnadviseSink(ByVal ConnectId As Long)

End Sub
```

Figure 6.13: Available Visual Basic Functions

6.6 Accessing Sunrise Clinical Manager Data

The code sample in Figure 6.14 shows how to access the data elements provided by the SCM shell. The available data elements and their names are listed in Table 6.15.

Table 6.15: Data Element Table (1 of 2)

Record Name	Description
DBDRIVER	The name of the database driver. This sometimes is needed to establish a connection to the database.
DBSERVER	The name of the database server to which this session of SCM is connected. This sometimes is needed to establish a connection to the database.
DBNAME	The name of the database to which this session of the SCM is connected. This sometimes is needed to establish a connection to the database.
PATIENTID	The patient MRN number for the current visit as shown in the Patient Header.
VISITNUMBER	The visit number for the current visit as shown in the Patient Header.
PATIENTGIVENNAME	The patient's given name as shown in the Patient Header.

This page contains Eclipsys proprietary information and is not to be duplicated or disclosed to unauthorized persons.

Table 6.15: Data Element Table (2 of 2)

Record Name	Description
PATIENTMIDDLENAME	The patient's middle name as shown in the Patient Header.
PATIENTLASTNAME	The patient's last name as shown in the Patient Header.
PATIENTLOCATIONFACILITYID	The facility of the current visit's assigned location.
PATIENTLOCATIONUNIT	The unit of the current visit's assigned location.
PATIENTLOCATIONROOM	The room of the current visit's assigned location.
PATIENTLOCATIONBED	The bed of the current visit's assigned location.
SESSIONID	The ID of the current context session. If there are multiple sessions of SCM running on the desktop, each has its own session ID. This is value is useful if connecting to the context outside of a custom tab.
USERID	The ID of the current user. This is the value the user types in at logon.
USERNAME	The user's full name in the format Last, Given Middle.
PROVIDERID-xxx	A list of the care provider IDs for the current user, where xxx is the care provider ID type. For example, PROVIDERID-PRIMARY would be the care provider ID of the type Primary. If the user does not have any care provider IDs, this entry does not exist.

In Figure 6.14, all the present name/value pairs are printed to the immediate window in debug mode.

```
Options Explicit
Implements IPageView
Dim mySink as IPageViewSink

Private Function IPageView_AdviseSink(ByVal PageViewSink _
 As BASICTABLib.IPageViewSink) As Long
Set mySink = PageViewSink
End Function
Private Sub IPageView_OnChangeVisit()
GetContextData
End Sub

Private Sub GetContextData()
Dim RecordNames() As String
Dim RecordValues() As String
Dim x as Integer

RecordNames = mySink.GetPatientRecordNames
RecordValues = mySink.GetPatientRecordValues(RecordNames)

For x = LBound(RecordNames) To UBound(RecordNames)
        Debug.Print RecordNames(X) & vbTab & RecordValues(X)
Next x
```


This page contains Eclipsys proprietary information and is not to be duplicated or disclosed to unauthorized persons.

```
End Sub
To access the data for one or more specific attributes, use the
following format:

RecordNames(0) = "PATIENTLOCATIONUNIT"
RecordNames(1) = "PATIENTLOCATIONFACILITY"
RecordValues = mySink.GetPatientRecordValues(RecordNames)

SUnit = RecordValues(0)
SFacility = RecordValues(1)
```

Figure 6.14: Accessing SCM Shell Data Elements

6.7 Controlling Tabbing Behavior

The following four functions allow the user to use the keyboard to alternate the focus between the tab itself and a control within the ActiveX control on the tab:

- SetFocusFirst.
- SetFocusLast.
- IsFirstWnd.
- IsLastWnd.

If the focus is on the tab itself and the user presses the Tab key, SetFocusFirst is called, giving the ActiveX control the opportunity to move the focus to the first control in the tab order.

If the focus in on the tab itself and the user presses Shift+Tab, SetFocusLast is called, giving the ActiveX control the opportunity to move the focus to the last control in the tab order.

When the user presses the Tab key, SCM asks the ActiveX control if the focus is currently on the last control in the ActiveX tab order, using IsLastWnd. If the ActiveX control responds TRUE, the focus is set to the tab itself (the tab label gets the focus rectangle).

When the user presses Shift+Tab, SCM asks the ActiveX control if the focus is currently on the first control in the ActiveX tab order using IsFirstWnd. If the ActiveX control responds True, the focus is set to the tab itself.

If an ActiveX control answer is True, no response is needed because True is the default response. If the control answer is False, control raises an error (see the sample below).

6.7.1 Example

Figure 6.15 code example assumes that txtText1 is the first control in the tab order and that cmdCancel is the last one.

```
Dim bIsFirstFocus as Boolean
Dim bIsLastFocus as Boolean
```


This page contains Eclipsys proprietary information and is not to be duplicated or disclosed to unauthorized persons.

```
Private Sub cmdCancel_GotFocus()

bIsLastFocus = True

End Sub

Private Sub cmdCancel_LostFocus()

bIsLastFocus = False

End Sub

Private Sub txtText1_GotFocus()

bIsFirstFocus = True

End Sub

Private Sub txtText1_LostFocus()

bIsFirstFocus = False

End Sub

Private Sub IPageView_IsFirstWnd()

If bIsFirstFocus = False Then
 Err.Raise vbObjectError + 601
End If

End Sub

Private Sub IPageView_IsLastWnd()

If bIsLastFocus = False Then
 Err.Raise vbObjectError + 602
End If
End Sub

Private Sub IPageView_SetFocusFirst()
'The following assumes that the control txtText1
 'is the first control in the tab order.
txtText1.SetFocus
End Sub
```

Figure 6.15: Controlling Tabbing Behavior Code Example

The alternative method in Figure 6.16 achieves the same result:

```
Private Sub IPageView_IsFirstWnd()

If ActiveControl <> txtText1 Then
        Err.Raise vbObjectError + 400    ' anything to generate
 error
End If

End Sub
```


This page contains Eclipsys proprietary information and is not to be duplicated or disclosed to unauthorized persons.

```
Private Sub IPageView_IsLastWnd()

If ActiveControl <> cmdCancel Then
        Err.Raise vbObjectError + 400    ' anything to generate
 error
End If

End Sub

Private Sub IPageView_SetFocusFirst()
'The following assumes that the control txtText1
 'is the first control in the tab order.
txtText1.SetFocus
End Sub

Private Sub IPageView_SetFocusLast()
'The following assumes that the control cmdCancel
 'is the last control in the tab order.
cmdCancel.SetFocus
End Sub
```

Figure 6.16: Controlling Tabbing Behavior Code Example Alternative Method

6.8 Obtaining Data From the Sunrise Clinical Manager Database

While it is not possible to update data in the SCM database from an ActiveX control hosted on a custom tab, it is possible to read and display data from the database. This data might be combined with other data from a different database for presentation to the user, or it might be used to create an HL7 transaction that could then be passed back into SCM using a link to an interface engine.

6.8.1 Security Considerations

To access data from the database, establish a connection to the database and log on using an appropriate SQL Server logon account. You cannot log on using the current SCM user's account, even if prompting the user for their password. This deliberate security feature prevents access to the database by a SCM user from outside the SCM application itself.

If direct access is allowed to the database from an external application, it is highly recommended setting up an SQL Server account that has Read access to only those tables needed to accomplish the task. Implementing and enforcing security from outside the SCM application is entirely the enterprise's responsibility. In this circumstance, the SCM application does not and cannot enforce any security rules. However, if the application limits access to only the patient found in the context, the application inherits the user's chart access permissions. For the patient context to be set the user must access the patient from within SCM; therefore, the SCM patient permissions for that user are used.

This page contains Eclipsys proprietary information and is not to be duplicated or disclosed to unauthorized persons.

If the application needs to know what rights a user has before presenting data for display, the current user's SCM rights are available for access. Sample SQL is provided in the next section.

6.8.2 Connecting to the Database

To connect to the SCM database, the application needs:

- A valid SQL Server user account.
- That user's SQL Server account password (not the SCM password).
- The database server name.
- The database name.

A sample OLEDB type of connection is shown below. Other types of connections are also possible. The section of the code that relates to references to mySink is described in more detail in "Accessing Sunrise Clinical Manager Data" on page 124.

```
Dim cnn1 As ADODB.Connection
Dim sNames as String
Dim sServer as String
Dim sDatabase as String
Dim sProvider as String
Dim sUser as String
Dim sPassword as String

'The database server and database names are obtained from the
context
    sNames(0) = "DBSERVER"
    sNames(1) = "DBNAME"
    sNamesValues = mySink.GetPatientRecordValues(sNames)
    sServer = sNamesValues(0)
    sDatabase = sNamesValues(1)
    sProvider = "SQLOLEDB"
    sUser = "insert appropriate username here"
    sPassword = "insert appropriate password here"

    ConnectToDatabase sServer, sProvider, sDatabase, sUser,
sPassword

Public Sub ConnectToDatabase(Server As String, Provider As String,
_
                DatabaseName As String, User As String, Password As
String)

        'This connection to the database uses ADO 2.0
       'you must add a reference to Microsoft ActiveX Data Objects
2.0
        'to this project

        ConnectionString = "Data Source=" & Server & ";" _
                           & "Provider=" & Provider & ";" _
                           & "Database=" & DatabaseName & ";" _
                           & "UID=" & User & ";" _
```


This page contains Eclipsys proprietary information and is not to be duplicated or disclosed to unauthorized persons.

```
                                & "PWD=" & Password & ";"

        Set cnn1 = New ADODB.Connection
        cnn1.Open ConnectionString

End Sub
```

Figure 6.17: Connection to the Database Code Example

6.8.3 Obtaining Patient Data

In SCM every row in every database table has a unique identifier called a GUID, which is also the table's primary key. Using a GUID is the most efficient method of retrieving data from the database. Every table that contains patient data has a column for the ClientGUID (the internal unique identifier for the patient) and almost all tables that contain patient data also have columns for the ChartGUID and the VisitGUID. The first step in getting patient data in all but the most simple of queries is to first obtain the various GUIDs (Client, Chart, and Visit). This allows for more efficient queries. If the GUIDs are stored in variables, they need to be obtained only once at the time the patient context changes.

The following sample code demonstrates how these GUID values can be obtained. The GUIDs are returned from the function GetKeyGUIDs as an array, with the first value being the ClientGUID, the second the ChartGUID, and the third the VisitGUID.

```
Option Explicit
Dim SQlStatement As String
Dim SELECTStatement As String
Dim FROMStatement As String
Dim WHEREStatement As String
Dim quote As String
Dim space As String
Dim rs1 As ADODB.Recordset

Public Function GetKeyGUIDs(PatientID As String, VisitID As
String, _
                LookupIDType As Boolean, FacilityCode As String, _
                Optional PatientIDType As String, _
                Optional VisitIDType As String) As Variant

'This function gets the GUIDs for the patient, chart and visit
'   based on the patient id and visit id that can be obtained from
the
'   context.  Since Sunrise Clinical Manager can support multiple
'   patient and visit id types, the combination of patient and
visit
'   id is not necessarily unique - it is the combination of id and
id
'   type that make them unique.  The id type is not present in the
'   context however, the id types can be determined from the
facility
'   code since when a facility is defined in config tools, the
patient
'   and visit ID types are specified.
'This function looks up the ID type if the value of LookupIDType is
```


This page contains Eclipsys proprietary information and is not to be duplicated or disclosed to unauthorized persons.

```
'   TRUE, but this takes longer and may not be necessary.  If there
is
'   only one patient ID type is defined, a lookup is not needed.
Even
'   if there is more than one id type, a lookup may not be needed
if
'   ID numbers are still likely to be unique.  In this case, this
'   function checks the number of rows returned (there must be only
'   one returned), and if more than 1 is returned, it looks up the
'   ID type based on the facility code.  This takes longer than if
we
'   had looked up the facility code in the first place, but because
'   duplicates are very rare, it is faster in the long run to look
up
'   the facility code only when necessary.
On Error GoTo ErrorHandler

Dim PatientIDType As String
Dim VisitIDType As String
Dim GUIDList(0 To 2) As String
Dim PatientGUID As String
Dim ChartGUID As String
Dim VisitGUID As String

quote = Chr(39)
space = Chr(32)

'First we need to determine if we should look up the appropriate
'   ID types.  If the value of LookupIDType is TRUE, and one or
both
'   the ID types is missing, use the facility code to look up the
ID
'   types used by that facility.

If (PatientIDType = "" Or VisitIDType = "") And LookupIDType Then
    LookupIDTypes FacilityCode
End If

TryAgain:
'Sample SQL statement to get the 3 GUIDs
'SELECT CV3VisitID.VisitGUID,
'       CV3ClientVisit.ChartGUID,
'       CV3ClientVisit.ClientGUID
'FROM   CV3ClientVisit, CV3VisitID
'WHERE  CV3VisitID.VisitGUID = CV3ClientVisit.Guid
'   AND    CV3VisitID.ClientIDType = "some patient id type"
'   AND    CV3VisitID.ClientIDCode = "some patient id number"
'   AND    CV3VisitID.VisitIDType = "some visit id type"
'   AND    CV3VisitID.VisitIDCode = "some visit id number"

'The following builds the above SQL statement using the patient ID
and
'   visit ID parameters passed into the function
SELECTStatement = "SELECT CV3VisitID.VisitGUID, " _
                  & "CV3ClientVisit.ChartGUID, " _
                  & "CV3ClientVisit.ClientGUID"

FROMStatement = "FROM CV3ClientVisit, CV3VisitID"
```


This page contains Eclipsys proprietary information and is not to be duplicated or disclosed to unauthorized persons.

```
WHEREStatement = "WHERE CV3VisitID.VisitGUID = CV3ClientVisit.GUID
"

'If we don't have a VisitIDType or PatientIDType then create a
where
'   statement that does not use them
If PatientIDType <> "" Then
WHEREStatement = WHEREStatement & " AND CV3VisitID.ClientIDType = "
_
                & quote & PatientIDType & quote
End If
If VisitIDType <> "" Then
WHEREStatement = WHEREStatement & " AND CV3VisitID.VisitIDType = "
_
                & quote & VisitIDType & quote
End If

'This part of the where statement is always needed
WHEREStatement = WHEREStatement & " AND CV3VisitID.ClientIDCode = "
_
                & quote & PatientID & quote _
                & " AND CV3VisitID.VisitIDCode = " _
                & quote & VisitID & quote

SQlStatement = SELECTStatement & space _
                & FROMStatement & space _
                & WHEREStatement

Set rs1 = New ADODB.Recordset
rs1.ActiveConnection = cnn1
rs1.Source = SQlStatement
rs1.CursorType = adOpenStatic
rs1.Open
If rs1.RecordCount = 1 Then
    rs1.MoveFirst
    VisitGUID = rs1.Fields(0)
    ChartGUID = rs1.Fields(1)
    PatientGUID = rs1.Fields(2)

    GUIDList(0) = PatientGUID
    GUIDList(1) = ChartGUID
    GUIDList(2) = VisitGUID
ElseIf rs1.RecordCount > 1 Then
    'look up the id types using the facility code because we got
more
    '   than 1 row back and we must have only one.
        LookupIDTypes FacilityCode
        GoTo TryAgain
Else
    MsgBox "No records were returned from the for patient ID " _
        & PatientID & ", visit ID " & VisitID
End If

GetKeyGUIDs = GUIDList

Exit Function
ErrorHandler:
    MsgBox "Error: " & Err.Description
End Function
```


This page contains Eclipsys proprietary information and is not to be duplicated or disclosed to unauthorized persons.

```
Private Function LookupIDTypes(FacilityCode As String) As Variant
'Sample SQL statement to get ID types when the FacilityCode is
known
'SELECT ClientIDType, VisitIDType
'FROM   CV3Location
'WHERE  (LevelCode= some facility code)
'       AND (IsFacility=1)

SELECTStatement = "SELECT ClientIDType, VisitIDType"

FROMStatement = "FROM CV3Location"

WHEREStatement = "WHERE IsFacility=1 AND LevelCode=" _
                                    & quote & FacilityCode & quote

SQlStatement = SELECTStatement & space
SQlStatement = SQlStatement & FROMStatement & space
SQlStatement = SQlStatement & WHEREStatement

Set rs1 = New ADODB.Recordset
    rs1.ActiveConnection = cnn1
    rs1.Source = SQlStatement
    rs1.CursorType = adOpenStatic
    rs1.Open

If rs1.RecordCount > 0 Then
   rs1.MoveFirst
Else
    MsgBox "No records were returned from the data query for
       FacilityCode " _
        & FacilityCode
End If

PatientIDType = rs1.Fields(0)
VisitIDType = rs1.Fields(1)


End Function
```

Figure 6.18: Obtaining Patient Data Code Example

This next sample function demonstrates how the GUID values obtained above can be used to obtain specific visit data. Some of the values returned in this sample are also available in the context. The array returned from this function contains 11 data elements.

```
Public Function GetPatientVisitData(PatientGUID As String, _
VisitGUID As String) As Variant

Dim DataList(0 To 10) As String
Dim i As Integer


'Sample SQL to get the patient and visit data using the patient
GUID
'   and visit GUID from CV3ClientVisit and CV3Client.

'SELECT CV3Client.LastName,
```


This page contains Eclipsys proprietary information and is not to be duplicated or disclosed to unauthorized persons.

```
'       CV3Client.FirstName,
'       CV3Client.BirthYearNum,
'       CV3Client.BirthMonthNum,
'       CV3Client.BirthDayNum,
'       CV3Client.ReligionCode,
'       CV3ClientVisit.CurrentLocation,
'       CV3ClientVisit.TemporaryLocation,
'       CV3ClientVisit.VisitStatus,
'       CV3ClientVisit.AdmitDtm,
'       CV3ClientVisit.DischargeDtm
'FROM   CV3Client, CV3ClientVisit
'WHERE  CV3ClientVisit.ClientGUID = CV3Client.Guid

SELECTStatement = "SELECT CV3Client.LastName, " _
& "CV3Client.FirstName, " _
& "CV3Client.BirthYearNum, " _
& "CV3Client.BirthMonthNum, " _
& "CV3Client.BirthDayNum, " _
& "CV3Client.ReligionCode, " _
& "CV3ClientVisit.CurrentLocation, " _
& "CV3ClientVisit.TemporaryLocation," _
& "CV3ClientVisit.VisitStatus," _
& "CV3ClientVisit.AdmitDtm," _
& "CV3ClientVisit.DischargeDtm"

FROMStatement = "FROM CV3Client, CV3ClientVisit"

WHEREStatement = "WHERE CV3ClientVisit.ClientGUID = CV3Client.GUID
" _
& " AND CV3ClientVisit.GUID = " _
& VisitGUID

SQlStatement = SELECTStatement & space _
& FROMStatement & space _
& WHEREStatement

Set rs1 = New ADODB.Recordset
rs1.ActiveConnection = cnn1
rs1.Source = SQlStatement
rs1.CursorType = adOpenStatic
rs1.Open
If rs1.RecordCount > 0 Then
rs1.MoveFirst

For i = 0 To rs1.Fields.Count - 1
If IsNull(rs1.Fields(i).Value) Then
    DataList(i) = ""
Else
    DataList(i) = rs1.Fields(i)
End If
Next i
Else
MsgBox "No records were returned for patient GUID " _
& PatientGUID & ", visit GUID " & VisitGUID
End If

GetPatientVisitData = DataList
End Function
```


This page contains Eclipsys proprietary information and is not to be duplicated or disclosed to unauthorized persons.

Figure 6.19: Obtaining Specific Visit Data Code Example

6.8.4 Checking Sunrise Clinical Manager User Security Rights

Sometimes the application needs to know whether or not a user has a particular SCM right. Because you can also create your own rights in SCM, it can be possible to use SCM's security definition features to store other rights the application requires.

To check a user's rights, first obtain the USERID from the context. See "Accessing Sunrise Clinical Manager Data" on page 124 for more information. After that value is found, use the function below to determine whether or not the user has a particular right.

```
Public Function CheckSecurityRights(UserName As String,
RightsList() As String, _
     Optional AllRights As String = False) As Boolean

'The following query is used to determine if the user has the
rights in
'   the RightsList.  If AllRights is TRUE, the user must have all
the
'   listed rights in order for the function to return TRUE.  If
AllRights
'   is FALSE, the function will return TRUE if the user has any of
the
'   rights in the list.
'
'SELECT DISTINCT r.Name, r.Category, r.IsFacilityDefined
'FROM (CV3UserSecurityGroup usg
'    INNER JOIN ((CV3SecurityGroup sg
'        INNER JOIN CV3SecurityGroupRights sgr ON sg.GUID =
sgr.SecurityGroupGUID)
'        INNER JOIN CV3Rights r ON sgr.RightsGUID = r.GUID)
'        ON usg.SecurityGroupGUID = sg.GUID)
'    INNER JOIN CV3User u ON usg.UserGUID = u.GUID
'WHERE   u.IDCode= <username>
'    AND     usg.EffectiveDtm < Now()
'    AND     (usg.ExpiryDtm > Now() OR usg.ExpiryDtm Is Null);

Dim CurrentDate As String
Dim x As Integer
Dim NumRights As Integer
Dim WhereList As String

CurrentDate = Now

SELECTStatement = "SELECT DISTINCT  r.Category, r.Name,
r.IsFacilityDefined "
FROMStatement = "FROM (CV3UserSecurityGroup usg " _
                & "INNER JOIN ((CV3SecurityGroup sg " _
                & "INNER JOIN CV3SecurityGroupRights sgr " _
                    & "ON sg.GUID = sgr.SecurityGroupGUID) " _
                & "INNER JOIN CV3Rights r ON sgr.RightsGUID =
r.GUID) " _
                & "ON usg.SecurityGroupGUID = sg.GUID) " _
                & "INNER JOIN CV3User u ON usg.UserGUID = u.GUID "
```


This page contains Eclipsys proprietary information and is not to be duplicated or disclosed to unauthorized persons.

```
WHEREStatement = "WHERE u.IDCode = " _
                    & quote & UserName & quote & " " _
                & "AND usg.EffectiveDtm < " _
                    & quote & CurrentDate & quote _
                & "AND (usg.ExpiryDtm > " _
                    & quote & CurrentDate & quote _
                & "OR usg.ExpiryDtm Is Null)"

'Get the list of rights in the RightsList and concatenate them then
'   add the list to the WhereStatement, keep a count of rights for
later.

NumRights = 0
For x = LBound(RightsList) To UBound(RightsList)
    If WhereList = "" Then
        WhereList = " AND (r.Name = " & quote & RightsList(x) &
quote
        NumRights = 1
    Else
        If RightsList(x) > "" Then
        WhereList = WhereList & " OR r.Name = " & quote &
RightsList(x) & quote
        NumRights = NumRights + 1
        End If
    End If

Next x

WhereList = WhereList & ")"

SQlStatement = SELECTStatement & space _
                    & FROMStatement & space _
                    & WHEREStatement & space _
                    & WhereList

Set rsSecurityRights = New ADODB.Recordset
    rsSecurityRights.ActiveConnection = cnn1
    rsSecurityRights.Source = SQlStatement
    rsSecurityRights.CursorType = adOpenStatic
    rsSecurityRights.Open

'Next check to see how many rows were returned.  If the same number
of rows were
'   returned as there are rights in the RightsList, the user has
all of them.  If
'   no rows were returned the user does not have any of the rights.
CheckSecurityRights = False

If AllRights = False Then
    If rsSecurityRights.RecordCount > 0 Then
        CheckSecurityRights = True
    End If
Else
    If rsSecurityRights.RecordCount = NumRights Then
        CheckSecurityRights = True
    End If
End If
```


This page contains Eclipsys proprietary information and is not to be duplicated or disclosed to unauthorized persons.

```
End Function
```

Figure 6.20: User Security Rights Code Example

6.9 Adding New ActiveX Tab Entries to Sunrise Clinical Manager Environment Profiles

After developing an ActiveX control, tell SCM to add a chart section (tab) to the user interface and how to compose the custom tab. The two pieces of information that SCM needs are the caption for the custom tab and the program ID (ProgID) of the ActiveX control. The program ID of an ActiveX control is created from data supplied when compiling the Visual C++ or Visual Basic program. For example, if creating an ActiveX control that provides Internet browsing capability, but only gives access to an intranet, identify the tab with the label Intranet, and designate MedService.IntraViewer as the ProgID of your control. With this information, SCM can construct a tab labelled Intranet with a copy of your control working in the body of the tab.

To provide the caption and ProgID information using the environment profile, take these steps:

- Create a new section called ActiveXTab at the highest level of the environment profile.
- Add an entry for the custom tab to the ActiveXTab section.
- Add new entries to the ViewLibrary section of the environment profile. The ViewLibrary determines what type of tab is added and where it is positioned with respect to the other tabs. The libraries are numbered from 0, which is the leftmost tab.

Note: This next procedure shows how to add an ActiveXTab section to the environment profile if it does not already exist. If the section already exists, proceed with the second task, “To add an ActiveXTab entry for a new custom tab” on page 138.

◆ To add an ActiveXTab section to the environment profile

1. In the Configuration Tools window, click Dictionary Maintenance. The Dictionary Maintenance window opens.
2. In the Categories list, click Miscellaneous.
3. In the Dictionaries list, click Environment Profile.
4. Select any top-level section.
5. Click Add This Level. The Add Environment Profile window opens.
6. In the Profile Type field, select Section/Sub Section.
7. Enter a description for the section.

This page contains Eclipsys proprietary information and is not to be duplicated or disclosed to unauthorized persons.

8 In the Code field, enter ActiveXTab.

9 Click OK.

◆ To add an ActiveXTab entry for a new custom tab

1 Select the ActiveXTab section.

2 Click Add Sub Level. The Add Environment Profile window opens.

3 In the Profile Type field, select Entry.

4 In the Code field, enter TabCaptionN. (N represents the next consecutive number from any previous TabCaption entries. If no other TabCaption entries exist, use zero (0). For example, if this is the first TabCaption entry, enter TabCaption0.)

5 In the Value field, enter the caption that identifies the custom tab.

6 Click OK.

7 Select the ActiveXTab section.

8 Click Add Sub Level. The Add Environment Profile window opens.

9 In the Profile Type field, select Entry.

10 Enter ProgIDN. (N represents the next consecutive number from any previous ProgID entries. If no other TabCaption entries, exist, use zero (0). For example, if this is the first ProgID entry, enter ProgID0.) The N value for ProgID and TabCaption must be the same and are treated as a pair.

11 In the Value field, enter the ProgID string identifying the ActiveX control to appear in the body of the custom tab.

12 Click OK.

◆ To add new ViewLibrary entries

1 In the Configuration Tools window, click Dictionary Maintenance. The Dictionary Maintenance window opens.

2 In the Categories list, click Miscellaneous.

3 In the Dictionaries list, click Environment Profile. The Environment Profile Maintenance window opens.

4 Select View Library.

5 Click Add Sub Level. The Add Environment Profile window opens.

6 In the Profile Type field, select Entry.

This page contains Eclipsys proprietary information and is not to be duplicated or disclosed to unauthorized persons.

7 In the Code field, enter ViewLibraryN. (N represents the next consecutive number from any previous ViewLibrary entries. For example, if the highest ViewLibrary entry is ViewLibrary6, enter ViewLibrary7.)

8 In the Value field, enter BasicTab.dll.

9 Click OK.

10 Select the ViewLibraryCount entry.

11 Click Edit.

12 In the Value field, increase the value by 1 (for example, if the value is 7, enter 8.)

13 Click Close.

This procedure assumes that adding the custom tab as the last chart section in the row is the goal. If the tab should be put between two existing chart sections, edit the values in the appropriate ViewLibrary entries for the tabs to reorder.

This page contains Eclipsys proprietary information and is not to be duplicated or disclosed to unauthorized persons.

This page contains Eclipsys proprietary information and is not to be duplicated or disclosed to unauthorized persons.

CHAPTER 7: AUTHENTICATION OPTIONS

Contents

7.1 Overview 141
7.2 Authentication Options. 141
7.3 Configuring Authentication 145
7.4 Authentication in a Citrix Environment 150
7.5 Special Authentication Conditions 151

7.1 Overview

Authentication is not the same as user authorization. Authentication only identifies the user; it does not authorize what a user can do.

Sunrise Clinical Manager allows for single factor authentication, as well as two-factor or strong authentication. SCM implements two forms of strong authentication; Username/password + Identix Biometric and RSA.

Combining any two of these methods is referred to as two-factor or strong authentication. SCM implements two forms of strong authentication; one based on knowledge of something and a unique characteristic and another based on knowledge of something and possession of something.

7.2 Authentication Options

SCM supports the following user authentication options.

- Username/Password.
- Biometric authentication (support for the Identix biometric fingerprint device only).
- Two-Factor Username/Password + Biometric.
- RSA SecurID Token/PIN.

This page contains Eclipsys proprietary information and is not to be duplicated or disclosed to unauthorized persons.

7.2.1 Username/Password Authentication

In this example, entering a username and password controls the authentication. The SCM logon window is shown in Figure 7.1.

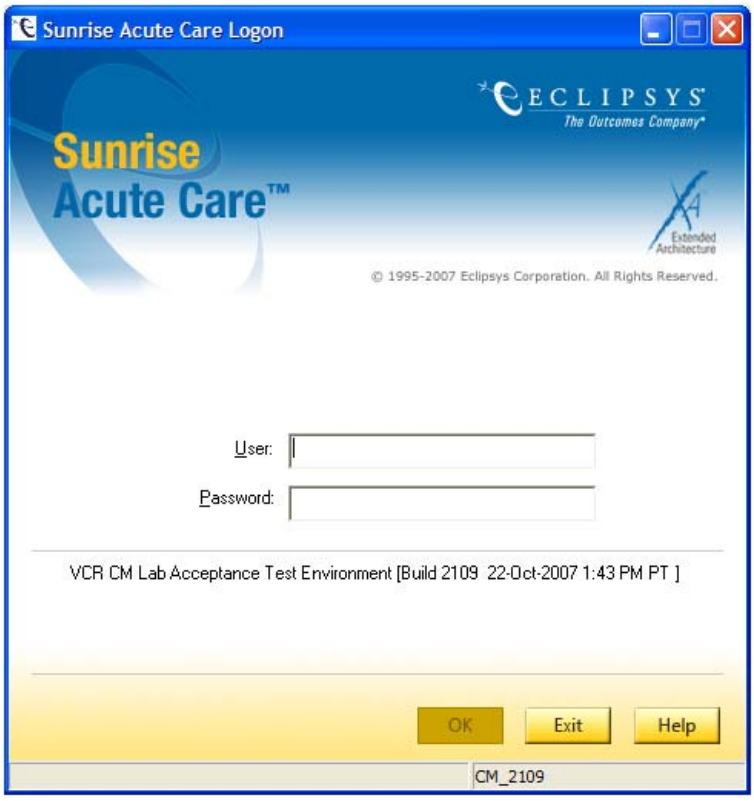


Figure 7.1: Logon Window

Note: Using Eclipsys Security Services there can also be a domain name in the logon window if Enterprise LDAP Server is used.

When Username/Password authentication is used, the following occurs:

1 During start-up, the SCM Logon window appears.

2 The user enters his username and password.

3 The user clicks OK.

The user is authenticated against the SCM user table.

7.2.2 Biometric Authentication

Biometric authentication is controlled by the Identix biometric fingerprint reader. An example of the Biometric Authentication window is shown in Figure 7.2.

This page contains Eclipsys proprietary information and is not to be duplicated or disclosed to unauthorized persons.



Figure 7.2: Biometric Authentication Window

When biometric authentication is the authentication configuration, the following occurs:

1 The user is presented with the Identix BioLogon window.

2 The user places the finger identified during the fingerprint enrollment process in the reader.

3 The reader scans the fingerprint, and the Identix system looks for a fingerprint match within the Identix credentials database.

4 After it is authenticated, a valid SCM User name is returned to the system and passed along to the authorization system before user access is granted to SCM.

7.2.3 Identix Biometric Requirements

Identix authenticates against valid domain users using ADS or LDAP Domains. This means that all users of SCM must be members of the domain. In addition, the SCM user name must be the same as the user's domain ID.

In order to authenticate correctly, users complete a biometric enrollment process. The user's fingerprint is read by the biometric device and recorded against his domain record. SCM requires that users be enrolled with the biometric system prior to using the SCM authentication process.

7.2.4 Two-Factor Username/Password + Identix Biometric

With Two-Factor authentication, SCM prompts the user to supply his username and password. The user is authenticated using this information. If successful, the Identix window appears and prompts the user to place his fingertip in the fingerprint reader. The Identix authentication system attempts to match the username to the fingerprint using the Identix data store. The following flowchart shows the Two-Factor authentication process.

This page contains Eclipsys proprietary information and is not to be duplicated or disclosed to unauthorized persons.

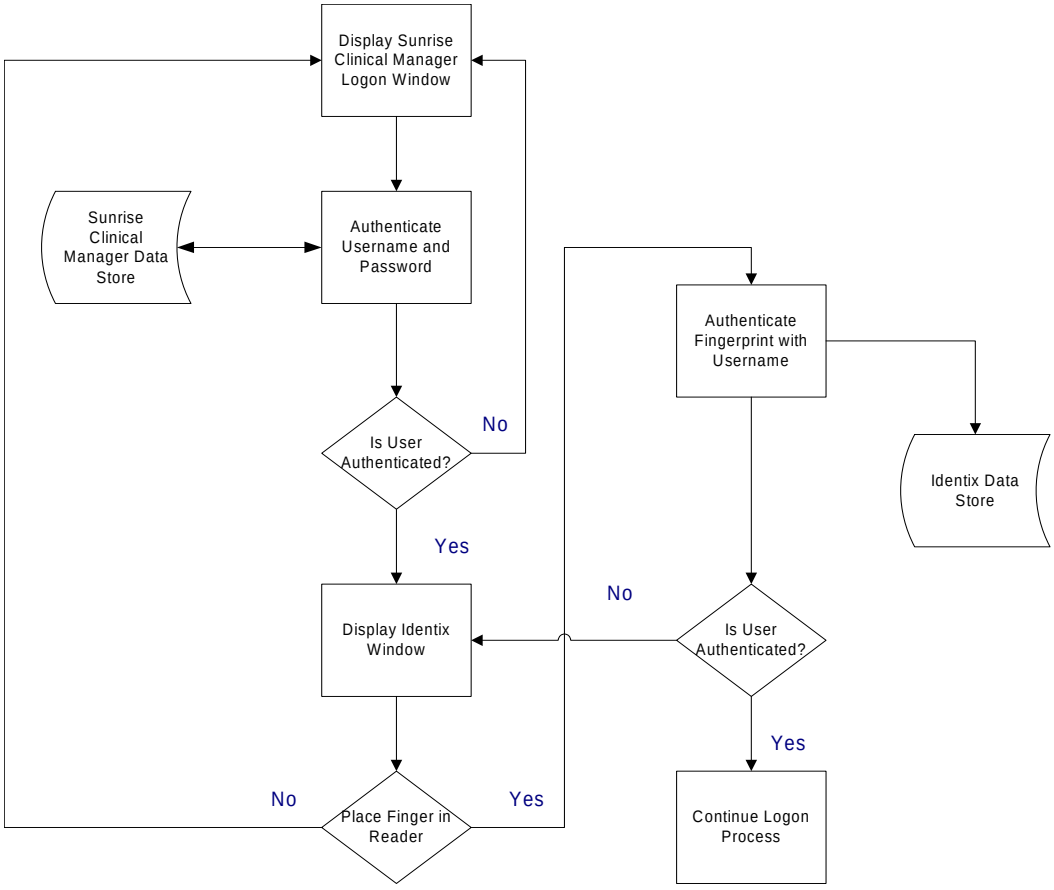


Figure 7.3: Two Factor Authentication Process

7.2.5 RSA SecurID

By default, using a SecurID is two-factor authentication, because it combines something known, such as the user's PIN with something the user has, such as the Smart Token.

The user is presented with the RSA SecurID logon (shown in Figure 7.4) where their SCM Username and RSA Passcode are entered.

The RSA Passcode is a combination of the user's PIN and the token number appearing on the RSA SecurID.

For example if the user's PIN is 5790 and the RSA SecurID displays 5835792, the RSA Passcode is 57905835792. The token number dynamically changes every 60 seconds.

This page contains Eclipsys proprietary information and is not to be duplicated or disclosed to unauthorized persons.



Figure 7.4: Client Passcode Verification Window

7.3 Configuring Authentication

SCM allows the enterprise to globally configure authentication and re-authentication options. Individual workstations can be configured to allow the user to override the global configuration.

7.3.1 Logon Authentication Options

Configure the SCM authentication options in the Authentication Option environment profile, which is found under HVCGlobalWorkstation>Authentication Configuration. The Authentication Option entry contains a numeric value to represent the authentication methods to use during the logon process.

The authentication options are global, but can be overridden at the workstation level by adding workstation specific environment profile entries. See the *Sunrise Clinical Manager General Configuration guide*, Environment Profiles for these entries.

7.3.2 Related Environment Profiles: Authentication Option

The following environment profile settings are part of the setup required for authentication options.

Set the values in the Environment Profile Maintenance dictionary (Dictionary icon > Dictionary Maintenance window > Miscellaneous category > Environment Profile dictionary).

This page contains Eclipsys proprietary information and is not to be duplicated or disclosed to unauthorized persons.

Table 7.1: Environment Profile Settings

Section > Entry	Default Value	Description	
HVCGlobalWorkstation > Authentication Configuration > Authentication Option	1	The authentication option used for the logon process. Valid values are 1, 2, 3, or 4.	
		Value	**Option Description**
		1	Username/Password
		2	Identix Biometric authentication
		3	Two-Factor Username/ Password + Identix Biometric
		4	RSA SecurID

7.3.3 Re-Authentication Options

Re-authentication options are used to inform SCM which methods to employ to re-authenticate the user. For example, the options indicate the method to use when a physician must re-authenticate to place an order within SCM. Configure the SCM authentication options in the Re-Authentication Option environment profile. The re-authentication options are global, but can be overridden at the workstation level by adding workstation specific environment profile entries. See the *Sunrise Clinical Manager General Configuration guide*, Environment Profiles for these entries.

The re-authentication options are identical to the logon options, it is not necessary to set them the same as the logon authentication. For example, the logon options can be set for Two-Factor Username/Password + Biometric, while the re-authentication option can be set to Biometric authentication, so only the biometric device is used to re-authenticate the physician when he places an order.

Note: Re-authentication from suspend session is the same as the logon authentication option configuration.

Configure the SCM re-authentication option in the Re-Authentication Option environment profile, which is found under HVCGlobalWorkstation>Authentication Configuration. The Re-Authentication Option entry contains a numeric value to represent the authentication methods used to re-authenticate the user.

This page contains Eclipsys proprietary information and is not to be duplicated or disclosed to unauthorized persons.

7.3.4 Related Environment Profiles: Re-Authentication Option

The following environment profile settings are part of the setup required for re-authentication options. Set the values in the Environment Profile Maintenance dictionary (Dictionary icon > Dictionary Maintenance window > Miscellaneous category > Environment Profile dictionary).

Table 7.2: Environment Profile Settings

Section > Entry	Default Value	Description	
HVCGlobalWorkstation > Authentication Configuration > Re-Authentication Option	1	The authentication option used for the logon process. Valid values are 1, 2, 3, or 4.	
		Value	**Option Description**
		1	Username/Password
		2	Biometric authentication
		3	Two-Factor Username/ Password + Biometric
		4	RSA Smart Token/PIN

7.3.5 Re-authentication Trigger Events

The available re-authentication trigger events are defined as environment profile entries under the HVCGlobalWorkstation node.

The following environment profile settings are part of the setup required for authentication options. Set the values in the Environment Profile Maintenance dictionary (Dictionary icon > Dictionary Maintenance window > Miscellaneous category > Environment Profile dictionary).

Table 7.3: Environment Profile Settings (1 of 2)

Section > Entry	Default Value	Description
HVCGlobalWorkstation > Documents > SignOnSubmit > New Document	FALSE	Should the password window pop up when new documents are submitted?
HVCGlobalWorkstation > Documents > SignOnSubmit > Sign	FALSE	Should the password window pop up when the document is being signed?
HVCGlobalWorkstation > Documents >SignOnSubmit > Flowsheets	FALSE	Should the password window pop up when the document is signed?

This page contains Eclipsys proprietary information and is not to be duplicated or disclosed to unauthorized persons.

Table 7.3: Environment Profile Settings (2 of 2)

Section > Entry	Default Value	Description
HVCGlobalWorkstation > Orders > SignOnSubmit > CancelDiscontinue	FALSE	Should the password window pop up when the order is canceled or discontinued?
HVCGlobalWorkstation > Orders > SignOnSubmit > Modify	FALSE	Should the password window pop up when the order is modified?
HVCGlobalWorkstation > Orders > SignOnSubmit > NewOrder	FALSE	Should the password window pop up when new orders are submitted?
HVCGlobalWorkstation > Orders > SignOnSubmit > ReleaseHold	FALSE	Should the password window pop up when a held order is released?
HVCGlobalWorkstation > Orders > SignOnSubmit > Sign	FALSE	Should the password window pop up when the order is being signed?
HVCGlobalWorkstation > Orders > SignOnSubmit > Suspend	FALSE	Should the password window pop up when the order is suspended?
HVCGlobalWorkstation > Orders > SignOnSubmit > Unsuspend	FALSE	Should the password window pop up when the order is unsuspended?
HVCGlobalWorkstation > Orders > SignOnSubmit > Verify	FALSE	Should the password window pop up when the order is to be verified?
HVCGlobalWorkstation > Tasks > Authentication > Sign	FALSE	Should the password window pop up when the task is being signed?

7.3.6 Workstation-Specific Authentication Options

Although the authentication and re-authentication options are global configurations, workstations can be configured to allow users to select from a list of allowable authentication and re-authentication methods for that workstation.

7.3.7 Alternate Authentication

A workstation can be configured to allow both biometric fingerprint authentication or RSA SecurID authentication; the user can select which method to employ during the logon process.

If more than one authentication option is defined and the UserCanSelectAuthentication environment profile is set to TRUE, a list of

This page contains Eclipsys proprietary information and is not to be duplicated or disclosed to unauthorized persons.

available authentication methods appears in the alternate window. The default authentication option is based on what has been specified in the HVCGlobalWorkstation > Authentication Configuration > Authentication Option, or Re-Authentication Option.

The enterprise can configure the text that appears for the authentication options in the alternate window using the Display Strings environment profile.

7.3.8 Related Environment Profiles: Alternate Authentication Options

The following environment profile settings are part of the setup required for alternate authentication options. Set the values in the Environment Profile Maintenance dictionary (Dictionary icon > Dictionary Maintenance window > Miscellaneous category > Environment Profile dictionary).

Table 7.4: Environment Profile Settings (1 of 2)

Section > Entry	Default Value	Description	
HVCGlobalWorkstation > Authentication Configuration > Alternate Authentication Option	1	The valid alternate authentication option used for the logon process. Valid values are 1, 2, 3, or 4 and are separated by commas.	
		Value	**Option Description**
		1	Username/Password
		2	Identix Biometric authentication
		3	Two-Factor Username/ Password + Identix Biometric
		4	RSA SecurID
HVCGlobalWorkstation > Authentication Configuration > Alternate Re-Authentication Option	1	The valid alternate authentication option used for the logon process. Valid values are 1, 2, 3, or 4 and are separated by commas.	
		Value	**Option Description**
		1	Username/Password
		2	Biometric authentication
		3	Two-Factor Username/ Password + Biometric
		4	RSA Smart Token/PIN

This page contains Eclipsys proprietary information and is not to be duplicated or disclosed to unauthorized persons.

Table 7.4: Environment Profile Settings (2 of 2)

Section > Entry	Default Value	Description
HVCGlobalWorkstation > Authentication Configuration > UserCanSelectAuthentication	FALSE	If more than one alternate authentication method is defined, indicates if the user can select from the available methods. TRUE/FALSE
HVCGlobalWorkstation > Authentication Configuration > Display Strings > RSA Token	RSA Token	RSA Token string displayed by alternate authentication window
HVCGlobalWorkstation > Authentication Configuration > Display Strings > Two Factor	Two Factor	Two-Factor string displayed by alternate authentication window
HVCGlobalWorkstation > Authentication Configuration > Display Strings > Biometric	Biometric	Biometric string displayed by alternate authentication window
HVCGlobalWorkstation > Authentication Configuration > Display Strings > User Password	User Password	User password string displayed by alternate authentication window
HVCGlobalWorkstation > DefaultToUsername PasswordonEPError	TRUE	Determines whether the User and Password is used if there is an error in the Environmental Profile entries related to Authentication.

7.4 Authentication in a Citrix Environment

In a Citrix environment, additional steps are needed if Biometric authentication or re-authentication is to be used.

◆ To enable strong authentication using Citrix on a Client Workstation

Note: These instructions are configured on the Citrix Client workstation.

1 Locate `VDSCMAUTHN.DLL` in the folder `Miscellaneous\BioMetrics` and unzip it in the Citrix client directory, typically `C:\Program Files\Citrix\ICA Client`.

2 Modify the `module.ini` file located in `C:\Program Files\Citrix\ICA Client` as follows:

- Add the following section immediately after the Smart Card Virtual Driver section:

This page contains Eclipsys proprietary information and is not to be duplicated or disclosed to unauthorized persons.

```
;*************************************************************
;**
;** SCM Virtual Driver
;**
;** Not available on Windows 16-bit machines and DOS.
;**
;*************************************************************
[SCMAUTH]
DriverName          = Unsupported
DriverNameWin16     = Unsupported
DriverNameWin32     = VDSCMAUTHN.DLL
Add SCMAUTH to the VirtualDriver listing in the ICA 3.0
WinStationDriver section:

VirtualDriver       = Thinwire3.0,  ClientDrive,
ClientPrinterQueue, ClientPrinterPort, Clipboard, ClientComm,
ClientAudio, ClientManagement, LicenseHandler,
ProgramNeighborhood,TWI,ZL_FONT,ZLC,SmartCard,ICACTL, SCMAUTH
Add SCMAUTH            = to the VirtalDriver listing in the Virtual
Driver section:

[VirtualDriver]
  Thinwire3.0         =
  ClientDrive         =
  ClientPrinterQueue  =
  ClientPrinterPort   =
  Clipboard           =
  ClientComm          =
  ClientAudio         =
  TWI                 =
  ClientManagement    =
  LicenseHandler      =
  ProgramNeighborhood =
  ZL_FONT             =
  ZLC                 =
  SmartCard           =
  ICACTL              =
  SCMAUTH             =
```

Figure 7.5: Enabling Strong Authentication on a Citrix Server

3 On the Citrix Server, set WideAreaCitrixModeEnabled Environmental Profile entry to TRUE in `HVC Global Workstation > Authentication Configuration`.

Note: The WideAreaCitrixModeEnabled environment profile entry is ignored if overridden in a workstation profile.

7.5 Special Authentication Conditions

7.5.1 Workstation Not Configured With Biometric Device

If a particular workstation does have access to a biometric device, the individual workstation can have workstation-specific environment profile entries to set different authentication options from the enterprise. These workstation-specific authentication options override the global authentication options.

This page contains Eclipsys proprietary information and is not to be duplicated or disclosed to unauthorized persons.

For more information, see the *Sunrise Clinical Manager General Configuration guide*, Environment Profiles for these entries.

7.5.2 Workstation Biometric Device Not Working

If the biometric device experiences a hardware failure, the workstation cannot be used to authenticate the user using biometrics until the biometric device is repaired. If the enterprise is allowed to use other methods of authentication, the workstation can be configured to support the username and password logon option until the biometric device is working.

7.5.3 Remote User Authentication

RSA SecurID is recommended for remote user authentication over Citrix because there are no client-side components to install on the user's workstation.

This page contains Eclipsys proprietary information and is not to be duplicated or disclosed to unauthorized persons.

Index

A

ActiveTab, 15
ActiveX Custom Tab
  accessing Clinical Manager data, 124
  ActiveX control interfaces, 110
  adding a tab to Clinical Manager environment profiles, 137
  controlling tab behavior, 126
  IPageView AdviseSink, 111
  IPageView CanChangePatient, 116
  IPageView CanCloseTab, 118
  IPageView FireActionMenuItem, 115
  IPageView GetActionMenu, 114
  IPageView IsFirstWnd, 120
  IPageView IsLastWnd, 121
  IPageView OnChangeVisit, 113
  IPageView SetFocusFirst, 119
  IPageView SetFocusLast, 119
  IPageView UnadviseSink, 112
  IPageViewSink GetPatientRecordNames, 107
  IPageViewSink GetPatientRecordValues, 109
  obtaining Clinical Manager data, 128
  overview, 105
  shell data link interfaces, 106
  using C++, 121
  Visual Basic, 122
AddTool, 21
AddTrigger, 18
Adobe Reader, 3
AdviseSink, 111
AllowDataLossOnContextChange, 30
Application, 10
Application_onToolExecute(nMenuID), 23
Application_OnUserExiot(), 24
Application_OnUserLogoff(), 24
Authentication
  authentication options, 141
  biometric, 142
  Citrix Environment, 150
  configuring authentication, 145
  Identix biometric requirements, 143
  overview, 141
  RSA Smart Token/PIN, 144
  special authentication conditions, 151
  two-factor, 143
  Username/Password, 142
Automation Control Interface
  _Application
  ActiveTab, 15
  AddTool, 21
  AddTrigger, 18
  Application, 10
  Application_OnToolExecute(nMenutID), 23
  Application_OnUserExit( ), 24
  Application_OnUserLogoff( ), 24
  ChangePassword, 18
  EnableUIPasswordChange, 22
  FullName, 12
  Logoff, 17
  Logon, 16
  Name, 11
  Parent, 10
  Quit, 13
  RemoveTool, 22
  RemoveTrigger, 20
  SessionID, 14
  Visible, 13
  _Application interface, 8
  AllowDataLossOnContextChange, 30
  desktop management for hiding and displaying Sunrise Clinical Manager, 6
  desktop management of Clinical Manager tools menu items, 7
  HL7 desktop integration, 7
  HL7 desktop integration support, 25
  HL7desktopstate, 26
  IHL7desktop interface, 25
  interfaces, 8
  JoinHL7Desktop, 28
  LeaveHL7Desktop, 29
  managing logon operations, 6
  overview, 5
  patient list refreshment after third-party data updates, 7
  single unified login shared between desktop application, 5

This page contains Eclipsys proprietary information and is not to be duplicated or disclosed to unauthorized persons.

starting the correct version of Sunrise Clinical Manager, 8

B

BeginContextChanges, 44

C

CanChangePatient, 116
CanCloseTab, 118
ChangePassword, 18
Context Server
data representation, 32
enabling the context server, 34
objects, 57
Class CoContextManager, 57
Context Data, 57
overview, 31
session control, 33
starting Clinical Manager from the command line, 34
Context Server Installation and Configuration, 33
Context Server Interfaces, 36
BeginContextChanges, 44
CreateContextSession, 39
FreezeContextSession, 49
GetApplicationSessions, 42
GetApplicationSessionsAndPIDs, 43
GetContextItems, 52
GetContextValues, 53
GetCurrentContextToken, 38
IHVCContextData, 52
IHVCContextManager, 36
IHVCContextParticipant, 56
JointContextSession, 39
LeaveContextSession, 41
Participant_ContextChanged(nContext), 56
Participant_Ping(), 56
PublishContextChanges, 46
ReleaseOwnership, 48
ReleaseSession, 51
SetContextValues, 54
TakeOwnership, 47
CreateContextSession, 39

D

Desktop Integration
clinical link icons, 97
clinical link menu, 98
common terms and phrases, 78
configuring Clinical Manager context controller, 96
configuring Clinical Manager for desktop integration, 86
configuring exit for all participant applications, 94
configuring medical record numbers, 91
configuring the vergence context manager for view context support, 83
configuring tools menu actions, 90
enabling a shortcut key, 91
managing the clinical link, 97
modes of context sharing, 81
prerequisite tasks, 81
responding to context changes, 99
viewing the clinical link state, 97

E

EnableUIPasswordChange, 22

F

FireActionMenuItem, 115
FreezeContextSession, 49
FullName, 12

G

GetActionMenu, 114
GetApplicationSessions, 42
GetApplicationSessionsAndPIDs, 43
GetContextItems, 52
GetContextValues, 53
GetCurrentContextToken, 38

H

HL7DesktopState, 26

I

IHVCContextData, 52
IHVCContextParticipant, 56
IsFirstWnd, 120

This page contains Eclipsys proprietary information and is not to be duplicated or disclosed to unauthorized persons.

IsLastWnd, 121

J

JoinContextSession, 39
JoinHL7Desktop, 28

L

LeaveContextSession, 41
LeaveHL7Desktop, 29
Logoff, 17
Logon, 16

M

Media Viewer
- configuring for Sunrise PACS, 69
- configuring media display, 62
- adding to the registered media viewers dictionary, 63
- server registration, 62
- confirm media file type associations, 68
- creating a media viewer server, 72
- IHVCMediaViewer interface specification, 73
- server requirements, 72
- image database location, 68
- overview, 61
- using HL7 messages, 70

N

Name, 11

O

OnChangeVisit, 113

P

Parent, 10
Participant_ContextChanged(nContext), 56
Participant_Ping(), 56
PublishContextChanges, 46

Q

Quit, 13

R

Referencing the Classes and Interfaces, 35
ReleaseOwnership, 48
ReleaseSession, 51
RemoveTool, 22
RemoveTrigger, 20
rotating pages, 3

S

SessionID, 14
SetContextValues, 54
SetFocusFirst, 119
SetFocusLast, 119

T

TakeOwnership, 47

U

UnadviseSink, 112

V

Visible, 13

This page contains Eclipsys proprietary information and is not to be duplicated or disclosed to unauthorized persons.

This page contains Eclipsys proprietary information and is not to be duplicated or disclosed to unauthorized persons.