IMiS(R)/Storage Connector Manual

Version 9.5.1510

IMAGING SYSTEMS

Imaging Systems Inc.
Brnciceva 41g
Ljubljana
Slovenia

TABLE OF CONTENTS

1 INTRODUCTION 5
1.1 About this guide 5
2 GENERAL 5
2.1 Integration with Java and .NET applications 5
2.2 Objects on different archive servers 6
2.3 Security and high availability 6
2.4 Advanced features 7
2.5 Use in SOA architecture 7
2.6 Versions 7
3 SYSTEM REQUIREMENTS 9
3.1 Hardware 9
3.1.1 Minimum requirements 9
3.1.2 Recommended requirements 10
3.2 Software 11
4 ADMINISTRATION 12
4.1 Installation 12
4.1.1 Installation process for the .NET version 12
4.1.2 Installation process for the Java version 23
4.2 Start up and shut down 23
4.3 Upgrading 23
4.3.1 Upgrading process for the .NET version 24
4.3.2 Upgrading process for the Java version 24
4.4 Removal 24
4.4.1 Removal process for the .NET version 24
4.4.2 Removal process for the Java version 28
5 RUNNING THE PRODUCT 28
5.1 Components 29
5.2 Interface for IMiS®/ARChive Server 7 30
5.2.1 “StorageConnector” 30
5.2.2 “Storage” 33
5.2.3 “Document” 34
5.2.4 “AuditLog” 35
5.3 Interface for IMiS®/ARChive Server 9 36
5.3.1 “StorageConnector” 39
5.3.2 “IArchive” 41
5.3.3 “IDirectory” and “IDirectoryEntity” 42

5.3.4 “IEntityStub” 43
5.3.5 “IEntity”, “IClass”, “IFolder” and “IDocument” 44
5.3.6 “IReadOnlyProperty” and “IProperty” 47
5.3.7 “IReadOnlyContent”, “IContent” and “IContentPart” 48
5.3.8 “IRetention”, “IRetentionPolicyEntry”, “IRetentionPolicyContext” and “IDispositionHoldEntry” 50
5.3.9 “IReviewStub” 51
5.3.10 “IReview” 52
5.3.11 “AuditLog” 54
5.3.12 “AuditQuery” 55
5.4 Examples of use 56
5.4.1 Initializing IMiS®/Storage Connector 56
5.4.2 Finalizing IMiS®/Storage Connector 56
5.4.3 Examples for IMiS®/ARChive Server version 7 57
5.4.4 Examples for IMiS®/ARChive Server version 9 61
5.4.5 Logging IMiS®/Storage Connector 74
6 TROUBLESHOOTING 79
6.1 Problems using IMiS®/Storage Connector .NET version 79
6.1.1 Issues with references in a development project 79
6.2 Problems using IMiS®/Storage Connector Java version 80
6.2.1 Issues with references in a development project 80
6.2.2 Issues with unhandled errors 81
6.2.3 Problems opening a session between the server and a client 82
6.2.4 Problems with log editing rights 83
6.3 List of errors that may occur when using IMiS®/Storage Connector 84
6.3.1 Errors in IMiS®/ARChive Server 7 84
6.3.2 Errors for IMiS®/ARChive Server 9 88

TABLE OF IMAGES

Table of images appearing in the manual

Image 1: IMiS®/StorageConnector architecture 6
Image 2: Preparing to install 13
Image 3: Beginning the IMiS®/StorageConnector installation procedure 13
Image 4: Cancelling the IMiS®/StorageConnector installation procedure 14
Image 5: Reviewing and accepting the license agreement 14
Image 6: Customer information dialog box 15
Image 7: Choice between complete and custom installation 15
Image 8: Selecting the elements and location of IMiS®/StorageConnector installation 16
Image 9: Description of the installation element icons 16
Image 10: Selecting the destination folder 17
Image 11: Available disk space 17
Image 12: Selecting the elements of IMiS®/StorageConnector .NET Developer Edition installation 18
Image 13: Confirming settings to begin installation 19
Image 14: Installation progress bar 19
Image 15: Installation complete message 20
Image 16: Silent installation using msiexex.exe program 20
Image 17: Uninstalling IMiS®/ StorageConnector using Add/Remove Programs 25
Image 18: Confirming uninstallation 25
Image 19: Uninstallation progress bar 25
Image 20: Opening the IMiS®/StorageConnector program maintenance 26
Image 21: Selecting a program maintenance action for the IMiS®/StorageConnector 26
Image 22: Confirming IMiS®/StorageConnector uninstallation 27
Image 23: Uninstallation complete message 27

1 INTRODUCTION

1.1 About this guide

This IMiS®/StorageConnector guide is intended for administrators and application developers with adequate prior knowledge who require information about installing, configuring and administering the IMiS®/StorageConnector interface. It is also intended for integrating applications with IMiS®/ARChive Server.

To better understand how the software works, and to achieve a more in-depth overview of archive server functionalities, administrators, application developers and users can also consult our guides for IMiS®/Client and IMiS®/ARChive Server.

Application developers have at their disposal development documentation with a detailed description of the IMiS®/StorageConnector interface, which is a part of the *Developer Edition* installation package.

2 GENERAL

IMiS®/Storage Connector is an application program interface (API) for transferring objects (scanned documents and other files) between application servers and IMiS®/ARChive Server. Objects are delivered at the request of the application for saving and viewing archived objects to/from the archive server. The interface facilitates fast response times and high throughput even when large numbers of objects are interchanged simultaneously.

2.1 Integration with Java and .NET applications

IMiS®/Storage Connector is built around the widely used .NET and Java software environments.
It contains a wealth of programming objects with an easy-to-use interface (API).
This makes it possible to quickly design functionally advanced applications for accessing IMiS®/ARChive Server.
A binary protocol is used for communication with the archive server, which essentially speeds up communication and ensures high responsiveness and throughput. Users of applications (DMS, ERP, CRM, BPM ...) view the delivered objects in these applications' integrated browsers.

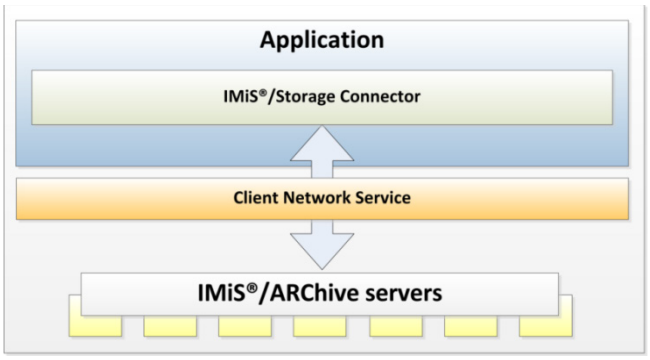

Image 1: IMiS®/StorageConnector architecture

IMiS®/Storage Connector .NET version is intended for integration with applications on .NET Framework 2.0, 3.5 and 4.0.

IMiS®/Storage Connector Java version is intended for integration with applications on J2EE 1.4.2_18 (or newer).

The .NET in Java versions of IMiS®/Storage Connector are available for IMiS®/ARChive Server version 7.

Only the .NET version of IMiS®/Storage Connector is currently available for IMiS®/ARChive Server version 9.

2.2 Objects on different archive servers

If a configuration with multiple IMiS®/ARChive Servers located at different locations is being used, users can access objects saved on multiple archive servers through a single application.

2.3 Security and high availability

IMiS®/Storage Connector uses an encrypted communications protocol and algorithms to communicate with different information systems. This prevents potential eavesdroppers without proper authorizations from accessing the information. It can also function in highly restrictive application environments which prevent access to the file system and which exclusively use executable memory (RAM) for their operations.

The use of automatic toggling between nodes provides IMiS®/ARChive Server with high availability in the event that problems occur. This ensures the 100% accessibility of archived content.

2.4 Advanced features

IMiS®/Storage Connector uses advanced algorithms for session pooling to ensure responsiveness and throughput for application servers with large workloads. For application solutions that require content streaming, the object model ensures all necessary components for these processes.

By using advanced caching algorithms it reduces the need for communication between application servers and the archive server and therefore provides the system with greater throughput.

2.5 Use in SOA architecture

IMiS®/Storage Connector can also be used with IMiS®/Storage Connector SOAP Service. This enables access to objects on IMiS®/ARChive Server through an online server as a web service. Communications between the interface and the archive server take place using a binary protocol, while communications with the application use a standardized service-oriented architecture (SOAP) protocol.
IMiS®/Storage Connector SOAP Service is used when support for an SOA communications protocol is needed (in terms of architecture or technology) to save and read objects on the archive server or to read object properties (time of creation, last edited, and other metadata about an object). It is currently only available for the Java environment and enables a web service to be set up on systems with J2EE 1.4.2_18 (or newer).

2.6 Versions

Product version labeling is based on a scheme that includes the following:

- An identifier of the installation platform (PLATFORM).
- An optional identifier of the processor architecture (ARCHITECTURE).
- Four separate numerical identifiers (MAJOR, MINOR, RELEASE, BUILD).
- An identifier for the installation package edition (EDITION).
- An installation package extension (EXTENSION) that changes based on the installation platform.

This is what the scheme looks like:

IMiS.StorageConnector.PLATFORM.ARCHITECTURE.MAJOR.MINOR.RELEASE.BUILD.EDITION.EXTENSION

The examples of installation package names for the .NET and Java platforms:

IMiS.StorageConnector.NET.x86.3.1.1210.Developer.Edition.msi
IMiS.StorageConnector.Java.3.1.1301.Runtime.zip

The scheme consists of the name of the IMiS®/StorageConnector module and the following elements:

- PLATFORM: This identifier represents the type of platform for which the installation package is intended.The possible values are *NET* and *Java*. NET represents an installation package intended for installation on the .NET platform, and Java represents an installation package intended for installation on the Java platform.
- ARCHITECTURE: This identifier represents the target processor architecture.
  The possible values are x86 and x64. x86 represents a 32-bit processor architecture, and x64 represents a 64-bit processor architecture.
- MAJOR: This identifier represents the major product version or product generation.
  It rarely changes in terms of changes in the system and features. A change indicates a considerable difference in the product compared to the previously released version.
  This identifier has a range of values from 1 to n; it is continuous and the values can only increase.
- MINOR: This identifier indicates a minor version of the product. It changes more frequently than the main version in terms of changes to the system, features and fixes.
  A change in the minor version represents smaller changes and fixes in the framework of the same product generation (indicated by the main or major version). The range of values is from 1 to n. This number is not continuous. It resets to its base value (1) with each new MAJOR version.
- RELEASE: This identifier represents the time component of the product release in accordance with the YYMM scheme.
  MM indicates the month of the release (range of values from 01 to 12), and YY indicates the last two digits of the year.
  Example: The RELEASE identifier for a product released in January of 2013 will read 1301.

- BUILD: The identifier in this position indicates the unique serial number of the product build; this number never repeats. If smaller changes are made to the product within a single month, this identifier may change. In this case, all other identifiers remain the same. The range of values is from 1 to n. This number is not continuous and can only increase.
- EDITION: This identifier represents the type of installation package based on the target users.
  Developer.Edition represents an installation package intended for developers.
  Besides libraries, this package also contains everything needed to develop applications that use the interface (development documentation, examples, etc.)
  Runtime indicates the installation package installed by an administrator. It contains the product libraries needed to run applications that use the product.

3 SYSTEM REQUIREMENTS

For successful installation and execution, the IMiS®/Storage Connector interface has the following hardware and software requirements.

3.1 Hardware

Practically all computers currently available on the market meet the hardware requirements for running IMiS®/Storage Connector.
Minimum and recommended requirements are listed below.

3.1.1 Minimum requirements

Minimum requirements for IMiS®/Storage Connector .NET:

- 400 MHz (.NET 2.0, 3.5) / 1 GHz (.NET 4.0) Intel Pentium 32-bit (x86) or 64-bit (x64) processor or other compatible processor.*
- 96 MB (.NET 2.0, 3.5) / 512 MB RAM (.NET 4.0) of RAM.*
- 5 MB (Runtime) / 650 MB (Developer Edition) of unused hard disk space.**
- Network access using the TCP/IP protocol (IPv4 or IPv6).

Notes:
** This is a brief summary of the minimum hardware requirements for .NET Framework 2.0, 3.5 and 4.0 as listed on Microsoft's website: http://msdn.microsoft.com/en-us/library/8z6watww%28v=vs.100%29.aspx*
*** The amount of unused space listed here is the space required for installation.*
*The IMiS®/Storage Connector .NET Developer Edition installation package includes installation packages for .NET Framework 2.0, 3.5 and 4.0. In case packages are not installed, additional amount of space is needed for installation (see the link under note *). Following installation, some of the space is freed up.*

Minimum requirements for IMiS®/Storage Connector Java:

- Intel Pentium 166 MHz processor.*
- 32 MB RAM.*
- 5 MB (Runtime) / 650 MB (Developer Edition) of unused hard disk space.**
- Network access using the TCP/IP protocol (IPv4 or IPv6).

Notes:
** This is a brief summary of the minimum hardware requirements for Java 2 Runtime Environment 1.4.2 for Microsoft Windows as listed on Oracle's website:*
http://www.oracle.com/technetwork/java/javase/install-windows-137451.html
*** The amount of unused space listed here is the space required for installation. Following installation, some of the space is freed up.*

3.1.2 Recommended requirements

Recommended requirements for IMiS®/Storage Connector .NET:

- 800 MHz (.NET 2.0, 1) / 1 GHz (.NET 4.0) Intel Pentium 32-bit (x86) or 64-bit (x64) processor or other compatible processor (or faster).*
- 256 MB (.NET 2.0, 3.5) / 512 MB (.NET 4.0) of RAM.*
- 5 MB (Runtime) / 650 MB (Developer Edition) of unused hard disk space.**
- Network access using the TCP/IP protocol (IPv4 or IPv6).

Notes:
** This is a brief summary of the minimum hardware requirements for .NET Framework 2.0, 3.5 and 4.0 as listed on Microsoft's website: http://msdn.microsoft.com/en-us/library/8z6watww%28v=vs.100%29.aspx*
*** The amount of unused space listed here is the space required for installation. The IMiS®/Storage Connector .NET Developer Edition installation package includes installation packages for .NET Framework 2.0, 3.5 and 4.0.*
*In case packages are not installed, additional amount of space is needed for installation (see the link under note *). Following installation, some of the space is freed up.*

Recommended requirements for IMiS®/Storage Connector Java:

- 166 MHz Intel Pentium processor or faster.*
- 48 MB or more of RAM.*
- 5 MB of unused hard disk space.**
- Network access using the TCP/IP protocol (IPv4 or IPv6).

Notes:
** This is a brief summary of the minimum hardware requirements for Java 2 Runtime Environment 1.4.2 for Microsoft Windows as listed on Oracles's website:*
http://www.oracle.com/technetwork/java/javase/install-windows-137451.html
*** The amount of unused space listed here is the space required for installation. Following installation, some of the space is freed up.*

3.2 Software

Software requirements for IMiS®/Storage Connector differ depending on the platform being used - .NET or Java.

Requirements for IMiS®/Storage Connector .NET:*

- Microsoft Windows 7 (32/64-bit), Windows Server 2003 (32/64-bit), Windows Server 2008 (32/64-bit).
- Microsoft .NET Framework 2.0, 3.5 or 4.0.

Notes:
** This is a summary of the operation systems supported for .NET Framework 2.0, 3.5 and 4.0 as listed on Microsoft's website: http://msdn.microsoft.com/en-us/library/8z6watww%28v=vs.100%29.aspx*

Requirements for IMiS®/Storage Connector Java:

- Microsoft Windows 7 (32-bit), Windows Server 2003 (32/64-bit), Windows Server 2008 (32-bit).
- Solaris 8, 9, 10 OS (32-/64-bit).
- Oracle Enterprise Linux 4.8, 5.4, 5.5, Red Hat Enterprise Linux AS 2.1 (32/64-bit), ES 2.1, WS 2.1, ES 3.0, AS 3.0 (32/64-bit), ES 4.0 (32/64-bit), AS 4.0 (32/64-bit), SUSE 8, 8.2, 9, 9.1, 9.2 (32/64-bit), 10, SLEC 8, SUSE Linux Enterprise Server 8 (32/64-bit), 9, 10, 11, TurboLinux 8.0, Sun Java™ Desktop System, Release 1, 2 .
- Java 2 Runtime Environment version 1.4.2_18

Notes:
** This is a summary of the operating systems supported for Java 2 Platorm 1.4.2 as listed on Oracle's website:*
http://www.oracle.com/technetwork/java/javase/system-configurations-139862.html

4 ADMINISTRATION

The IMiS®/Storage Connector interface can be administered by administrators and/or application developers.
Administration encompasses installation, start up, shut down, upgrading and removal of the software.

4.1 Installation

Installation can be performed by an administrator in an environment that meets the minimum installation requirements. The minimum requirements can be upgraded if the need to do so is foreseen.

IMiS®/Storage Connector .NET is available in two different MSI installation packages:

- IMiS®/Storage Connector .NET Runtime contains interface libraries that are installed to the Global Assembly Cache (GAC).
- IMiS®/Storage Connector .NET Developer Edition contains everything needed for the development, distribution and execution of applications that use the interface: libraries, examples for developers and tools for application redistribution and the option of installing libraries to the Global Assembly Cache (GAC).

IMiS®/Storage Connector Java is currently only available as a ZIP package containing libraries and interface development documentation. Installation of the Java package is performed manually and requires the installation of libraries to the appropriate location.

4.1.1 Installation process for the .NET version

IMiS®/Storage Connector .NET installation can be performed by an administrator using the installation package or manually. Installation with the installation package can be performed using a wizard or as a silent installation without a user interface. Installation using a wizard is performed in English.

4.1.1.1 Installation with the installation wizard

The process for installing IMiS®/Storage Connector .NET using the installation wizard is described below. The installation wizard is a user interface found in the installation package. It guides the administrator through the installation process.

An example of an installation package name:

IMiS.StorageConnector.NET.x86.3.1.1301.msi.

Installation begins by running the installation package from the file system. A dialog box will appear informing the administrator that the installation package is being prepared for installation.

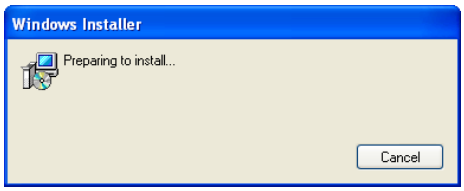

Image 2: Preparing to install

This is followed by the wizard's welcome screen, where the user can choose to continue or cancel the installation.

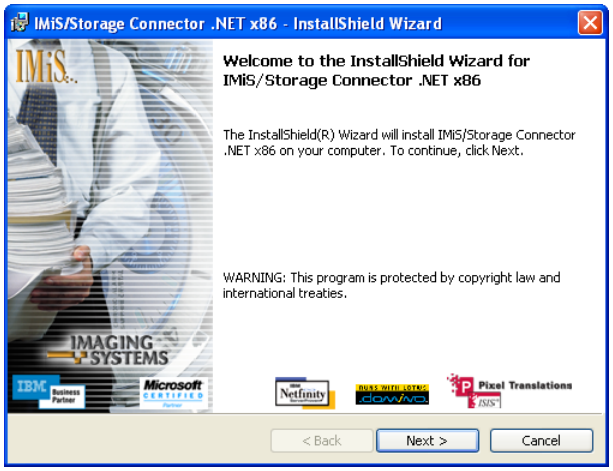

Image 3: Beginning the IMiS®/StorageConnector installation procedure

In each of the following steps, the user may perform the following actions:

- By clicking the Next button, they may continue the installation.
- By clicking the Back button, they may go back to the previous step.
- By clicking the Cancel button, they may cancel the installation.

If the installation process is cancelled using the *Cancel* button, a dialog box appears.
By selecting *Yes*, the user will cancel the installation; by selecting *No*, the user will continue with the installation process.

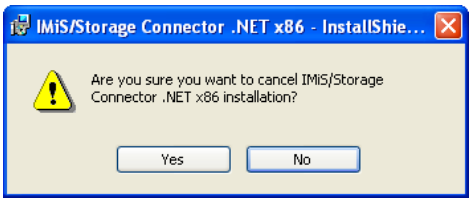

Image 4: Cancelling the IMiS®/StorageConnector installation procedure

If the installation process is cancelled, any installation files and settings installed in the Windows register are deleted.

In the following step, the administrator carefully reads the terms of the licensing agreement.
If the administrator agrees with the terms, they select "*I accept the terms in the license agreement*" and accept the licensing terms in their entirety.
If the administrator does not agree with the terms, they select "*I do not accept the terms in the license agreement*" and click the *Cancel* button to cancel the installation process.

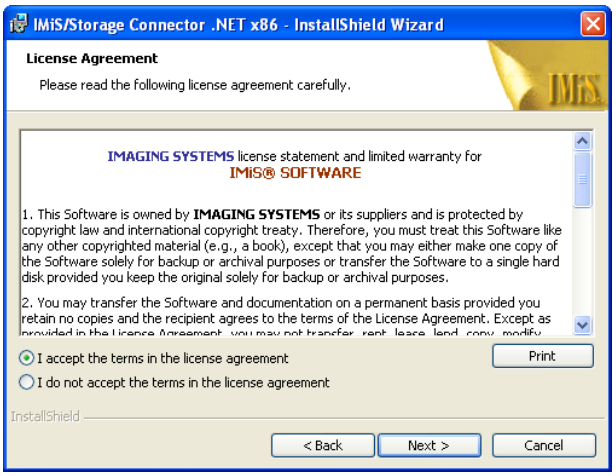

Image 5: Reviewing and accepting the license agreement

The installation process will then prompt the user to enter a user name in the *User Name* field and the name of the organization in the *Organization* field. They then select whether the application will only be installed for the current user - the "*Only for me*" option - or for all users on this computer - the "*Anyone who uses this computer*" option.

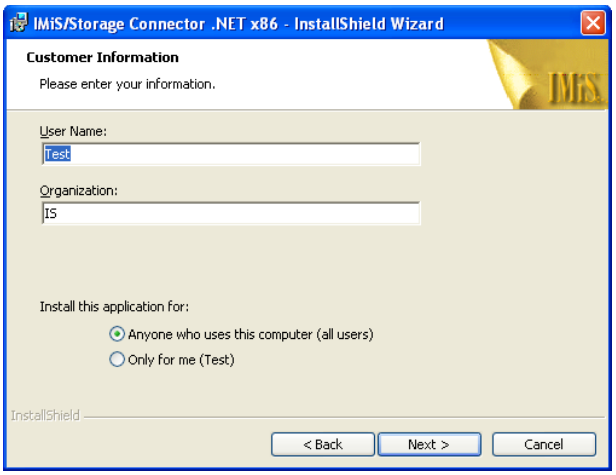

Image 6: Customer information dialog box

In the following step the user selects whether they would like to perform a complete installation (*the Complete option*) or a custom installation (*the Custom option*). Complete installation will install all files from the installation package on the file system.

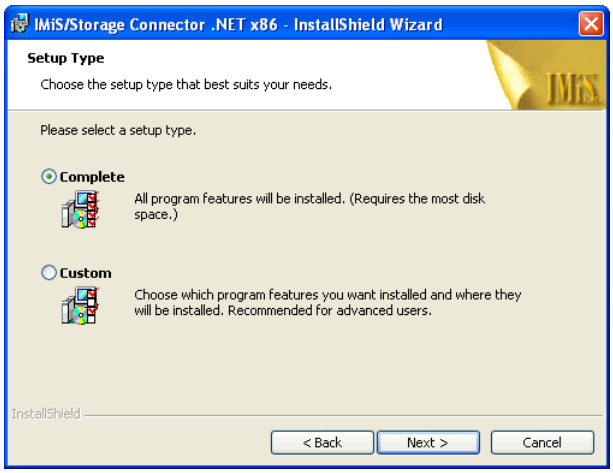

Image 7: Choice between complete and custom installation

If the user chooses custom installation, a dialog box will appear where they may select which IMiS®/Storage Connector .NET elements they would like to install and specify a destination.

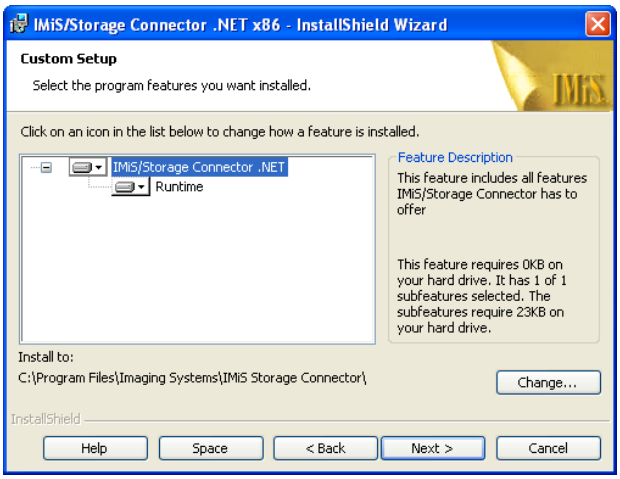

Image 8: Selecting the elements and location of IMiS®/StorageConnector installation

Clicking the *Help* button will open a window with information on the icons that appear in front of the names of the elements for installation.

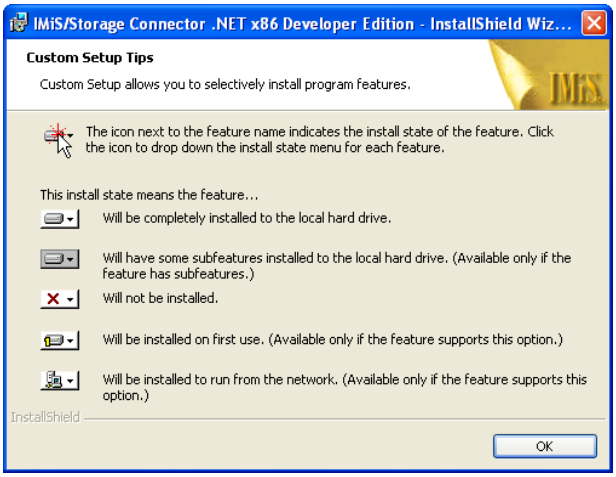

Image 9: Description of the installation element icons

By clicking the *Change* button, the administrator can change the installation location of IMiS®/Storage Connector .NET. A dialog will appear where they may select the destination folder; once selected, the selection is confirmed by pressing *OK*.

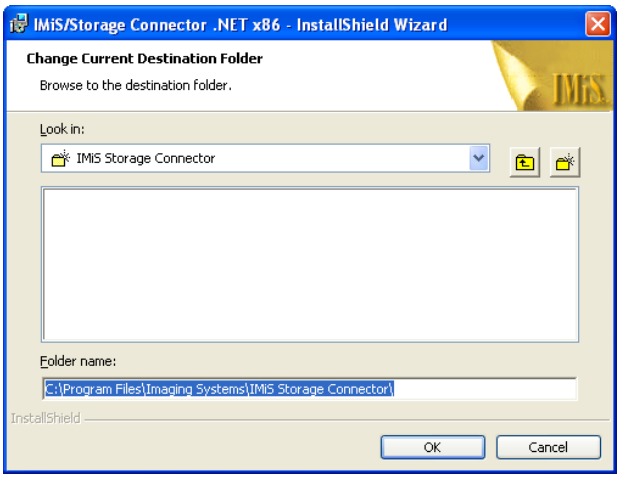

Image 10: Selecting the destination folder

Clicking the *Space* button will show whether there is enough space at the desired location. A dialog box will appear with a list of all available disks and information on their size and the space available on individual disks. Disks on which too little space is available for installation are highlighted.

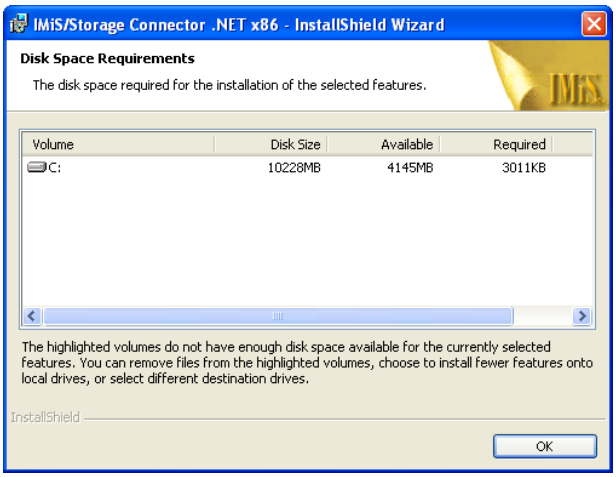

Image 11: Available disk space

The selection of elements available for installation depends on the version of the installation package.

With the IMiS®/Storage Connector .NET Runtime version, only the Runtime element is available. This element will install IMiS®/Storage Connector .NET to the Global Assembly Cache (GAC).

With the IMiS®/Storage Connector .NET Developer Edition version of the installation package, the administrator can choose from the following elements:

- *Development for Microsoft .NET v2.0*: this will install the environment for developing application with the IMiS®/Storage Connector .NET interface for .NET Framework 2.0. The environment contains the appropriate libraries, development documentation and examples and the installation package for .NET Framework 2.0.
- *Development for Microsoft .NET v3.5*: this will install the environment for developing application with the IMiS®/Storage Connector .NET interface for .NET Framework 3.5. The environment contains the appropriate libraries, development documentation and examples and the installation package for .NET Framework 3.5.
- *Development for Microsoft .NET v4.0*: this will install the environment for developing application with the IMiS®/Storage Connector .NET interface for .NET Framework 4.0. The environment contains the appropriate libraries, development documentation and examples and the installation package for .NET Framework 4.0.
- *Redistributables*: this will install tools for the redistribution of applications that use the IMiS®/Storage Connector .NET interface.
- *Runtime*: this will install IMiS®/Storage Connector .NET to the Global Assembly Cache (GAC).

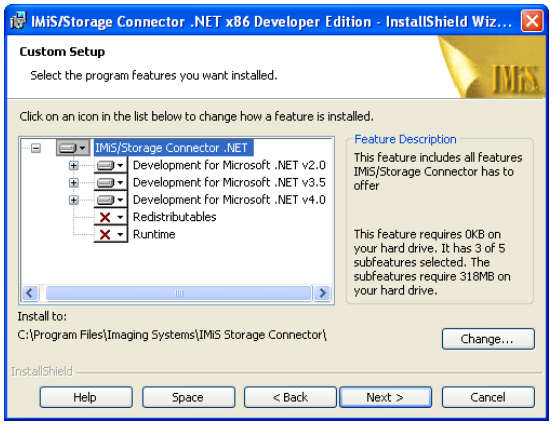

Image 12: Selecting the elements of IMiS®/StorageConnector .NET Developer Edition installation

In the following step of the installation wizard, the administrator will be asked to confirm the selected installation and can start the installation process by clicking the Install button.

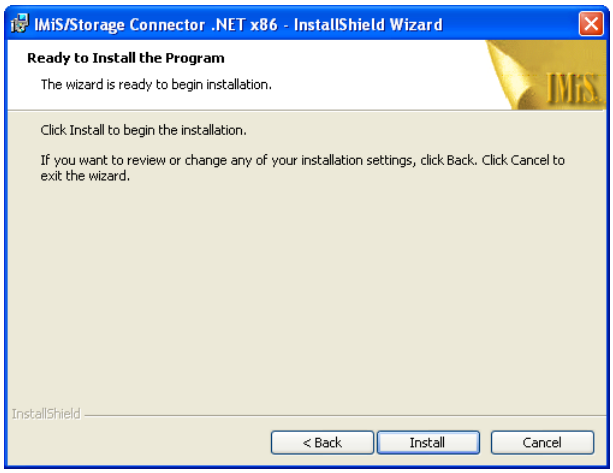

Image 13: Confirming settings to begin installation

This will start the process for installing IMiS®/Storage Connector .NET. The progress bar in the window shows how far along the process of transferring the files to the specified destination is. Installation can take from several seconds to several minutes, depending on the version of the installation package and the speed of the computer.

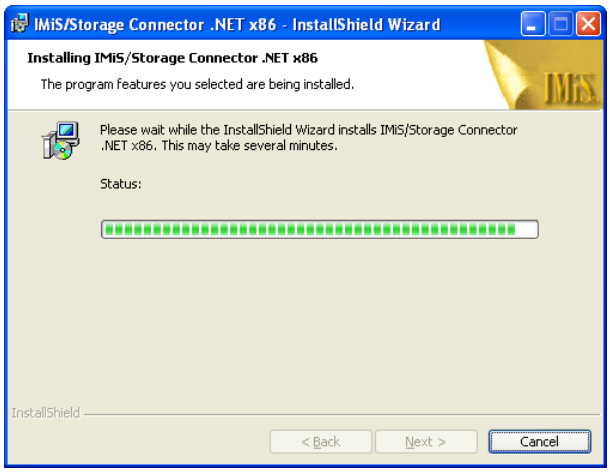

Image 14: Installation progress bar

Upon completion of installation, a dialog box will appear informing the administrator that installation has been successfully completed. To close the window, click the *Finish* button.

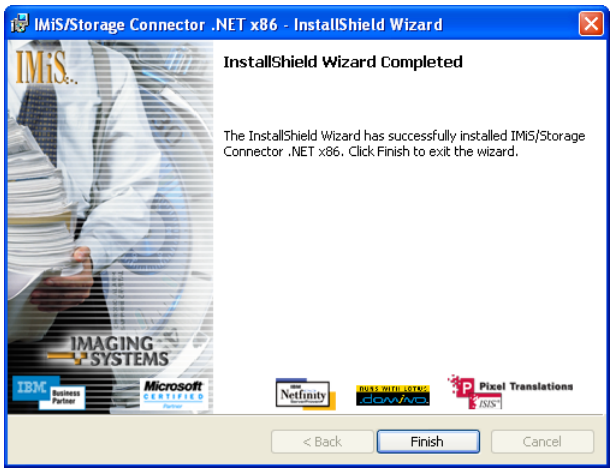

Image 15: Installation complete message

4.1.1.2 Silent installation

IMiS®/Storage Connector .NET installation can also be performed without user guidance or an installation wizard. This type of installation is known as silent installation.

To perform silent installation, the administrator uses the *msiexec.exe* program, which can be found in the *System32* Windows system directory. This tool forms part of Microsoft's installation product. It is used to perform various maintenance tasks on applications installed on the Windows operating system.

Additional information about *msiexec.exe* is available on Microsoft's website:

http://msdn.microsoft.com/en-us/library/windows/desktop/aa367449(v=vs.85).aspx

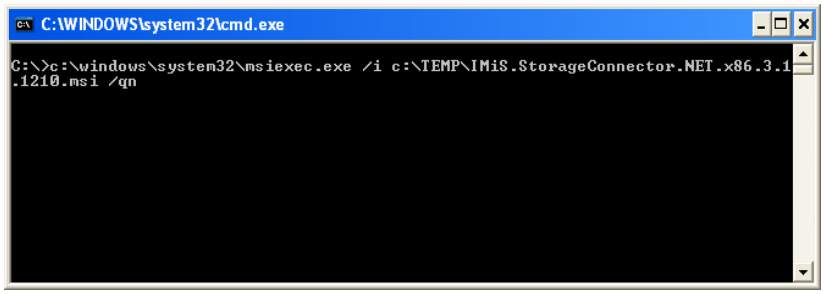

Image 16: Silent installation using msiexex.exe program

Silent installation of IMiS®/Storage Connector .NET can be launched using a command line in the *cmd.exe* command prompt, which is also located in the *System32* Windows system directory.

The command line consists of the *msiexec.exe* program, */i* parameters, which are used to specify the path to the installation package, and *q* and */qn* parameters, which are used to specify the installation method without the use of a user interface. Installation can take from several seconds to several minutes, depending on the version of the installation package and the speed of the computer.

An example of a command line for silent installation:

C:\>msiexec.exe /i c:\TEMP\IMiS.StorageConnector.NET.x64.3.1.1302.msi /qn

The table below lists common methods for installing with *msiexec.exe* which can be specified using the parameters. A list of all parameters is available on Microsoft's website: *http://msdn.microsoft.com/en-us/library/windows/desktop/aa367988(v=vs.85).aspx*

Parameters	Description
/q	Installation without the use of a user interface.
/qn	Installation without the use of a user interface. Same as /q.
/qn+	Installation without the use of a user interface, with a modal window upon completion of installation.
/qb	The basic user interface with a simplified progress bar. The /gb! parameter is used to hide the *Cancel* button.
/qr	A simplified user interface, without a modal window upon completion of installation.
/qf	The entire user interface, including all dialog boxes, a progress bar and a list of errors upon completion of installation.

The administrator can further customize silent installation of IMiS®/Storage Connector .NET using parameters specific to this installation. Parameters are added to the end of the command line in the following form: parameter=value.

An example of a command line for silent installation to the selected directory:

C:\>msiexec.exe /i c:\TEMP\IMiS.Scan.8.6.1211.Full.msi /qn INSTALLDIR=C:\IMIS

The table below contains descriptions of the supported command line parameters:

Parameters	Value	Description
INSTALLDIR	<directory name>	This value contains the default installation directory (Default = "%PROGRAMFILES%\IS\IMiS Storage Connector\").
USERNAME	<user name>	The value contains the user name of the user performing the installation (the default value is taken from the system settings).
COMPANYNAME	<company name>	The value contains the name of the company performing the installation (the default value is taken from the system settings).

Logging of silent installation can be turned on with the log parameter.

An example of a command line for silent installation with logging:

C:\>msiexec.exe /i c:\TEMP\IMiS.StorageConnector.NET.x64.3.1.1302.msi /log c:\TEMP\setup.log /qn

Additional information about *msiexec.exe* is available on Microsoft's website:

http://msdn.microsoft.com/en-us/library/windows/desktop/aa367449(v=vs.85).aspx

4.1.1.3 Manual installation

IMiS®/Storage Connector .NET can also be manually installed.

To install IMiS®/Storage Connector .NET Runtime, the administrator must install all libraries to the Global Assembly Cache (GAC). This is done with the *gacutil.exe* program.

This tool was installed together with Microsoft Visual Studio or Windows SDK.

The command line is made up of the *gacutil.exe* program and */i* parameters used to specify the path to the .NET library the user would like to install.

An example of a command line for installing libraries to the GAC:

C:\>gacutil.exe /i imisbase.net.dll
C:\>gacutil.exe /i iacxnone.net.dll
C:\>gacutil.exe /i iarcli.net.dll
C:\>gacutil.exe /i storageconnector.net.dll

Additional information about *gacutil.exe* is available on Microsoft's website:

http://msdn.microsoft.com/en-us/library/ex0ss12c(v=vs.80).aspx
http://msdn.microsoft.com/en-us/library/ex0ss12c(v=vs.90).aspx
http://msdn.microsoft.com/en-us/library/ex0ss12c(v=vs.100).aspx

To use IMiS®/Storage Connector .NET in a development environment, it is sufficient to copy all the libraries (*see chapter 5.1 Components*) to the selected folder and to create a reference to the *storageconnector.dll* library in the development project.

4.1.2 Installation process for the Java version

Installation of the IMiS®/Storage Connector Java interface is performed manually.
To install IMiS®/Storage Connector Java Runtime, all libraries need to be copied (*see chapter 5.1 Components*) to a special location in Java 2 Runtime Environment (J2RE) or Java 2 SDK (J2SDK) through which the *Java Extension Mechanism* will be able to locate the suitable JAR library.

This location is:

<java-home>/lib/ext *[in Java 2 Runtime Environment]*
<java-home>/jre/lib/ext *[in Java 2 SDK]*

Here *<java-home>* represents the folder where J2RE or J2SDK is installed.
Additional information about *Java Extension Mechanism* and installing JAR libraries is available on Oracle's website: *http://docs.oracle.com/javase/1.4.2/docs/guide/extensions/index.html*

To use IMiS®/Storage Connector Java in a development environment, it is sufficient to copy all the libraries (*see chapter 5.1 Components*) to the selected folder and to add a reference to the folder through
JAR-class-path. Additional information about how Java locates the suitable libraries is available on Oracle's website: *http://docs.oracle.com/javase/1.4.2/docs/tooldocs/findingclasses.html*

4.2 Start up and shut down

IMiS®/Storage Connector does not have a user interface. It is started through an application that uses the IMiS®/Storage Connector .NET or IMiS®/Storage Connector Java interface.
The administrator also shuts down the program through the application.

4.3 Upgrading

If an administrator would like to upgrade a past version of the IMiS®/Storage Connector interface to a newer version, they follow the steps described below.

4.3.1 Upgrading process for the .NET version

Before starting upgrading the administrator must make sure that any applications that use IMiS®/Storage Connector .NET are not being used. If any such applications are in use, they first need to be shut down.
The upgrading process depends on how the current version was installed - with an installation package or manually.

If the current version was installed using an installation package the administrator can perform the upgrade without first removing existing files. The new version is installed using the process described in *chapter 4.1.1.1 Installation with a installation wizard* and *chepter 4.1.1.2 Silent installation*. If an older version is installed, the installation package will automatically remove it and then install the new version.

If the installation was performed manually (as described in *chapter 4.1.1.3 Manual installation*) the old versions can be manually removed from the GAC using a command line (as described in *chapter 4.4.1.2 Manual removal*). Old versions of libraries installed in a development environment can simply be overwritten with the new libraries.

4.3.2 Upgrading process for the Java version

Before starting upgrading the administrator must make sure that any applications that use IMiS®/Storage Connector Java are not being used. If any such applications are in use, they first need to be shut down.
In the upgrading process, old versions of the libraries installed in the Java Extension directory or the development environment are simply overwritten with the new libraries (*see chapter 4.1.2 Installation process for the Java version*).

4.4 Removal

4.4.1 Removal process for the .NET version

Before removal the administrator must make sure that any applications that use IMiS®/Storage Connector .NET are not being used. This is done by checking that the application(s) that use these libraries are not running. The removal process then depends on how the IMiS®/Storage Connector interface was installed.

4.4.1.1 Removing the installation package

If IMiS®/Storage Connector was installed using an installation package and a wizard (*see chapter 4.1.1.1 Installation with the installation wizard*) or silent installation, it can be removed from the computer simply by using the *Add or Remove Programs* application in Windows.

This application can be found by clicking the *Start* button on the desktop and selecting *Add or Remove Programs* from the *Control Panel*.

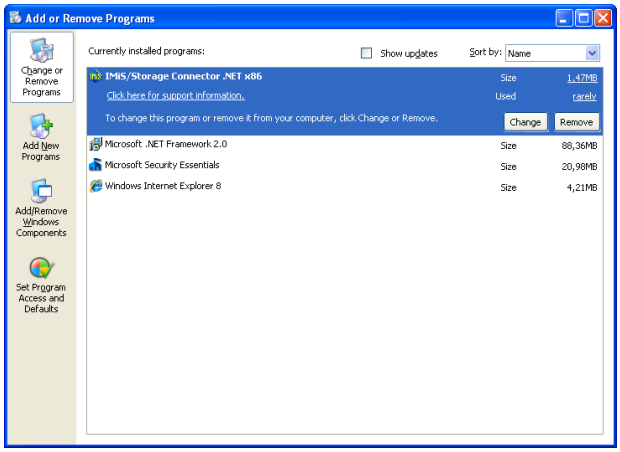

Image 17: Uninstalling IMiS®/ StorageConnector using Add/Remove Programs

Clicking the *Remove* button will open a dialog where the administrator confirms the removal by clicking *Yes* or cancels the removal by clicking *No*.

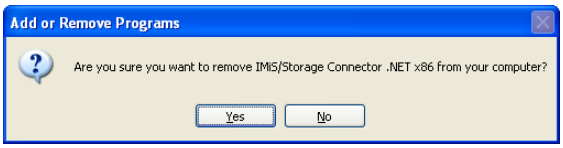

Image 18: Confirming uninstallation

If the removal is confirmed, it will start. Progress can be monitored in the dialog box.
To cancel the removal, click the Cancel button.

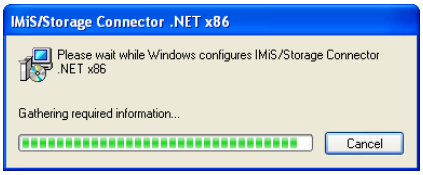

Image 19: Uninstallation progress bar

The removal process removes all files and settings created by the installation package.

The administrator can also remove the program by clicking the *Change* button.

This will first open the welcome screen of the installation wizard. The process can be continued by clicking the *Next* button or cancelled by clicking *Cancel*.

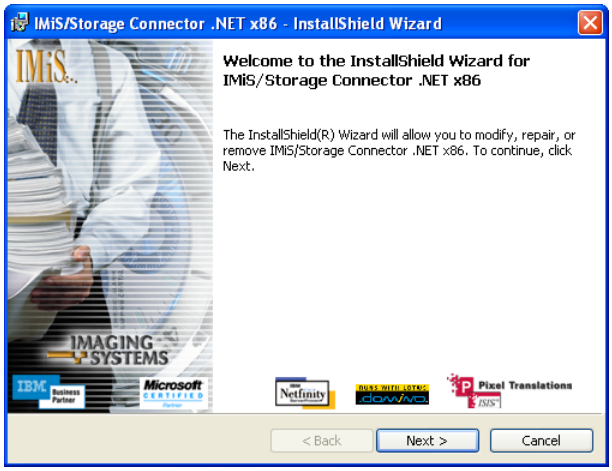

Image 20: Opening the IMiS®/StorageConnector program maintenance

Clicking *Next* will open a dialog box with multiple options. To remove the program, the administrator selects the *Remove* option and clicks *Next*.

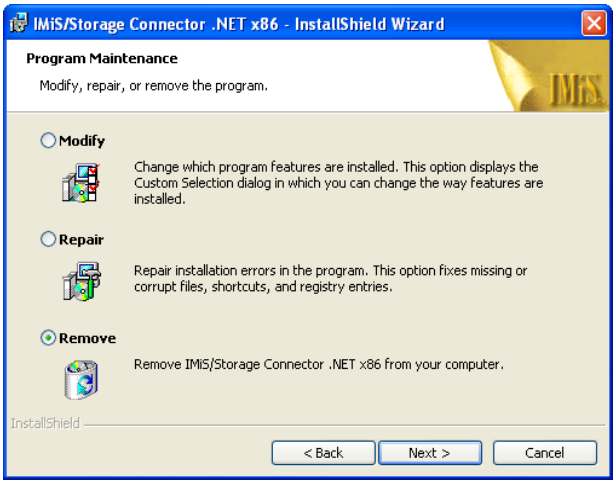

Image 21: Selecting a program maintenance action for the IMiS®/StorageConnector

In the next step, they confirm the removal by clicking the *Remove* button.

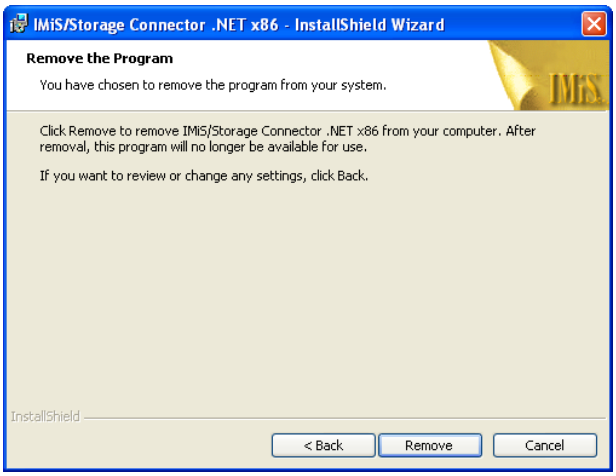

Image 22: Confirming IMiS®/StorageConnector uninstallation

Removal can take from several seconds to several minutes, depending on the version of the installation package and the speed of the computer. Once removal is complete, a final dialog window will appear. To close the window, click the Finish button.

Image 23: Uninstallation complete message

4.4.1.2 Manual removal

If IMiS®/Storage Connector was manually installed (*see chapter 4.1.1.3 Manual installation*) the administrator must also remove it manually.

If IMiS®/Storage Connector .NET Runtime is installed, the administrator must remove all libraries from the Global Assembly Cache (GAC). This is done with the *gacutil.exe* program, which is part of Microsoft Visual Studio or Windows SDK.

An example of a command line for removing libraries from the GAC:

C:\>gacutil.exe /u imisbase.net.dll
C:\>gacutil.exe /u iacxnone.net.dll
C:\>gacutil.exe /u iarcli.net.dll
C:\>gacutil.exe /u storageconnector.net.dll

If IMiS®/Storage Connector .NET is installed in a development environment, all libraries must be removed from the folder where they were installed.

4.4.2 Removal process for the Java version

To remove IMiS®/Storage Connector Java Runtime, all libraries installed to the special location in Java 2 Runtime Environment (J2RE) or Java 2 SDK (J2SDK) where the *Java Extension Mechanism* finds the suitable JAR library (*see chapter 4.1.2 Installation process for the Java version*) need to be deleted.

This location is:

<java-home>/lib/ext *[in Java 2 Runtime Environment]*
<java-home>/jre/lib/ext *[in Java 2 SDK]*

Here *<java-home>* represents the directory where J2RE or J2SDK is installed.

Removing IMiS®/Storage Connector Java in a development environment is equivalent to deleting all the libraries (*see chapter 5.1 Components*) from the selected folder in the development project (*see chapter 4.1.2 Installation process for the Java version*).

5 RUNNING THE PRODUCT

IMiS®/Storage Connector is used in applications or application servers to access IMiS®/ARChive Server. It runs on the .NET and Java platforms using an API.
The API is similar in both cases, as development was coordinated.

This chapter describes the basic components of the IMiS®/Storage Connector interface for the .NET and Java platforms and the most common examples of use of the API.
A more detailed description of the interface is available in the development documentation that forms a part of the *Developer Edition* installation package for the .NET and Java environments.

5.1 Components

The basic components of IMiS®/Storage Connector are libraries used by applications to access IMiS®/ARChive Server. With IMiS®/Storage Connector .NET version, these are assembly libraries in the form of *.dll* files.

With the IMiS®/Storage Connector Java version, they are equivalent to JAR libraries.

The main library which contains the *storageconnector.net.dll* interface in .NET or the *storageconnector.jar* interface in Java is described below.

All IMiS®/Storage Connector libraries are described in the table below:

.NET	
Library	**Description**
imisbase.net.dll	*IMiS Base Assembly* - contains the basic components used in other IMiS .NET libraries and applications.
iacxapi.net.dll	*IMiS/ARC Compression API Assembly* - contains components that form the basis for all compression libraries for IMiS®/ARChive Server.
iarcli.net.dll	*IMiS/ARC Client Assembly* – contains components that enable work with sessions and objects in IMiS®/ARChive Server.
storageconnector.net.dll	*IMiS/Storage Connector Assembly* – contains components that enable work with archives and documents on IMiS®/ARChive Server and other archive servers.
sl-SI\iarcli.net.resources.dll	*IMiS/ARC Client Resources Assembly* – contains translations of *IMiS/ARC Client Assembly* in the language specified by the folder where the library is located.*
sl-SI\storageconnector.net.resources.dll	*IMiS/Storage Connector Assembly* – contains translations of *IMiS/Storage Connector Assembly* in the language specified by the folder where the library is located.*

** Library not required to run IMiS®/Storage Connector .NET.*

Java	
Library	**Description**
imisbase.jar	*IMiS Base JAR* - contains basic components that implement frequently used objects and features in IMiS® Java libraries and applications.
iacxapi.jar	*IMiS/ARC Compression API JAR* - contains components that form the basis for all compression libraries for IMiS®/ARChive Server.
iarcli.jar	*IMiS/ARC Client JAR* – contains components that enable work with sessions and objects in IMiS®/ARChive Server.
storageconnector.jar	*IMiS/Storage Connector JAR* – contains components that enable work with archives and documents in IMiS®/ARChive Server and other archive servers.

5.2 Interface for IMiS®/ARChive Server 7

The application program interface (API) for IMiS®/ARChive Server version 7 consists of three objects in the *IMiS.StorageConnector* address space:

- *StorageConnector* – the main object of the IMiS®/Storage Connector application program interface.
- *Storage* – an archive on the client side linked to a specific archive server.
- *Document* – a document on the client side linked to a specific object on the archive server.

Besides logging, the *StorageConnector* object enables the opening of archives in the form of *Storage* objects. These objects in turn enable the opening of documents (files) in the archive represented by the *Document* objects. Besides these objects, the *AuditLog* object is also available for logging the audit trail for the documents in the archive.

A detailed description of interface objects for IMiS®/ARChive Server version 7 is presented below.

5.2.1 "StorageConnector"

StorageConnector is the primary object of the IMiS®/Storage Connector program interface. Access to the *singleton* object instance is managed through the *Instance* property, which is only valid until the *FreeInstance* method is called up. It contains methods for opening archives on different archive servers.

At the time of writing, only IMiS®/ARChive Server is supported. It enables logging settings and contains constants of the names of different options which are delivered together with the suitable values when opening an archive.

Below, the archive to IMiS®/ARChive Server is marked with the prefix *IMiSARC* for the .NET version and with the prefix *IMiS_ARC* for the Java version. This usage is in accordance with the StorageType constants in the interface.

The *StorageConnector* object contains the following elements:

.NET	
Constant	**Description**
MAX_OPTION_NAME_LENGTH	The maximum length of names for the audit trail.
OptionApplicationName	Option for the *IMiSARC* archive that determines the name of the application for the audit trail.
OptionAuthCryptoAlgorithm	Option for the *IMiSARC* archive that determines the type of cryptographic algorithm used in authentication.
OptionAuthCryptoCipherMode	Option for the *IMiSARC* archive that determines the type of operation with block ciphers in authentication.
OptionAuthCryptoKeySize	Option for the *IMiSARC* archive that determines the size of the authentication key.
OptionAuthKey	Option for the *IMiSARC* archive object that specifies the authentication key.
OptionAuthType	Option for the *IMiSARC* archive that determines the type of authentication.
OptionComputerName	Option for the *IMiSARC* archive that determines the name of the computer for the audit trail.
OptionMaxSessionsPerUser	Option for the *IMiSARC* archive that determines the maximum number of sessions per user.
OptionNodes	Option for the *IMiSARC* archive that specifies any additional archive servers that make up the cluster.
OptionObjectIdEncoding	Option for the *IMiSARC* archive that determines how an object identifier is encoded.
OptionObjectIdKind	Option for the the *IMiSARC* archive object that specifies what kind of (internal/external) object identifiers the program is dealing with.
OptionObjectIdKind	Option for the *IMiSARC* archive object that specifies what kind of (short/long) object identifiers the program is dealing with.
OptionUserName	Option for the *IMiSARC* archive object that determines the name of the user for the audit trail.

Property	**Description**
Instance	Returns a singleton instance of the *StorageConnector* object. The first time the program is used an instance is created. This instance is returned until the *FreeInstance* method is called up, at which point the singleton instance is no longer valid.
LogHandlers	Returns a list of external logging handlers.
LogInternal	Returns or sets a value that tells the user whether internal logging is performed or not.
LogLevel	Returns or sets the current level of logging.
LogLocale	Returns or sets regional settings for logging.
VersionInfo	Returns the IMiS®/StorageConnector version.

Method	**Description**
FreeInstance	Terminates the singleton instance of the *StorageConnector* object. Once this method is used, calling up the Instance property will not return a valid instance, as this instance no longer exists.
OpenContentManagerStorage	Opens an archive for IBM DB2 Content Manager Server.*
OpenContentManagerStorage	Opens an archive for IBM Lotus Domino Document Manager or a Domino.Doc-compatible archive server.*
OpenFileSystemStorage	Opens an archive for a local/remote file system.*
OpenIMiSARCStorage	Opens an archive for IMiS®/ARChive Server 7 (the *IMiSARC* archive) through the provided network address (host name or IP address) and port.
ContentTypeResolver	Makes it possible to get an extension from the content type (MIME) and vice versa.

** IMiS®/Storage Connector .NET does not currently support these features.*

Java	
Field	**Description**
MAX_OPTION_NAME_LENGTH	The maximum length of names for the audit trail.
OPTION_APPLICATION_NAME	Option for the *IMIS_ARC* archive that determines the name of the application for the audit trail.
OPTION_AUTH_CRYPTO_ALGORITHM	Option for the *IMIS_ARC* archive that determines the type of cryptographic algorithm used in authentication.
OPTION_AUTH_CRYPTO_CIPHERMODE	Option for the *IMIS_ARC* archive that determines the type of operation with block ciphers in authentication.
OPTION_AUTH_CRYPTO_KEYSIZE	Option for the *IMIS_ARC* archive that determines the size of the authentication key.
OPTION_AUTH_KEY	Option for the *IMIS_ARC* archive that specifies the authentication key.
OPTION_AUTH_TYPE	Option for the *IMIS_ARC* archive that determines the type of authentication.
OPTION_COMPUTER_NAME	Option for the *IMIS_ARC* archive that determines the name of the computer for the audit trail.
OPTION_MAX_SESSIONS_PER_USER	Option for the *IMIS_ARC* archive that determines the maximum number of sessions per user.
OPTION_NODES	Option for the *IMIS_ARC* archive that specifies any additional archive servers that make up the cluster.
OPTION_OBJECT_IDENCODING	Option for the *IMIS_ARC* archive that determines how an object identifier should be encoded.
OPTION_OBJECT_IDKIND	Option for the *IMIS_ARC* archive that specifies what kind of (internal/external) object identifiers the program is dealing with.
OPTION_OBJECT_IDTYPE	Option for the *IMIS_ARC* archive that specifies what type of (short/long) object identifiers the program is dealing with.
OPTION_USER_NAME	Option for the *IMIS_ARC* archive that determines the name of the application for the audit trail.

Method	**Description**
freeInstance	Terminates the singleton instance of the *StorageConnector* object. Once this method is used, calling up the *getInstance* property will not return a valid instance, as this instance no longer exists.
getInstance	Returns a singleton instance of the *StorageConnector* object. The first time the program is used an instance is created. This instance is returned until the freeInstance method is called up, at which point the singleton instance is longer valid.
getVersionInfo	Returns the IMiS®/StorageConnector version.
logAddHandler	Returns a list of external handlers for logging.
logGetLevel	Returns the current level of logging.
logInternal	Enables or disables internal logging.
logIsInternal	Returns a value that tells the user whether internal logging is performed or not.
logRemoveHandler	Removes the handler from the list of external handlers for logging.
logSetLevel	Sets the level of logging.
openContentManagerStorage	Opens an archive for IBM DB2 Content Manager Server.*
openDocumentManagerStorage	Opens an archive for IBM Lotus Domino Document Manager or a Domino.Doc-compatible archive server.*
openFileSystemStorage	Opens an archive for a local/remote file system.*
openIMiSARCStorage	Opens the archive for IMiS®/ARChive Server (*IMIS_ARC*) through the provided network address (host name or IP address) and port.
setMimeExtResolver	Sets an external object for determining an extension from the MIME type and vice versa.

** IMiS®/Storage Connector .NET does not currently support these features.*

5.2.2 "Storage"

The *Storage* object represents an archive on the client side linked to a specific archive server. It contains operations such as creating, opening, saving, handing off and deleting documents. A message for the audit trail report can also be set for certain objects using the *AuditLog* object (*see chapter 5.3.8 "Auditlog"*). Note that a message must be separately set for each operation.

The Storage object contains the following elements:

.NET	
Property	**Description**
AuditLog	Returns the *AuditLog* object for setting a message for the audit trail.
Capacity	Returns the size or capacity of the archive in bytes.*
IsClosed	Returns a value that tells the user whether the archive is closed or not.
ObjectCount	Returns the number of documents in the archive.*
SpaceUsed	Returns the amount of space used in bytes.*
StoreInfo	Returns information about the archive.
StoreType	Returns the type of archive.

Method	**Description**
Close	Closes the archive.
CreateObject	Creates a document in the selected profile.
DeleteObject	Deletes a document listed with an identifier from the archive.
GetProfileCapacity	Returns the size of the profile in bytes.*
GetProfileObjectCount	Returns the number of objects in the profile.*
GetProfileSpaceAvailable	Returns the amount of unused space in the profile in bytes.*
GetProfileSpaceUsed	Returns the amount of space used in the profile in bytes.*
MoveObject	Moves a document to the selected profile.
OpenObject	Opens a document in the archive.*
RetrieveObject	Retrieves a document from the archive to a file on the local disk or to the data stream.
StoreObject	Saves a document to the archive in the selected profile with the provided MIME type.

** IMiS®/Storage Connector .NET does not currently support these features.*

Java	
Method	**Description**
Close	Closes the archive.
createObject	Creates a document in the selected profile.
deleteObject	Deletes a document listed with an identifier from the archive.
getAuditLog	Returns the *AuditLog* object for setting a message for the audit trail.
getCapacity	Returns the size or capacity of the archive in bytes.*
getProfileCapacity	Returns the size of the profile in bytes.*
getProfileObjectCount	Returns the number of objects in the profile.*
getProfileSpaceAvailable	Returns the amount of unused space in the profile in bytes.*
getProfileSpaceUsed	Returns the amount of space used in the profile in bytes.*
getSpaceAvailable	Returns the amount of unused space in the archive in bytes.*
getSpaceUsed	Returns the amount of space used in the archive in bytes.*
getStoreInfo	Returns information about the archive.

** IMiS®/Storage Connector .NET does not currently support these features.*

Java (continued)	
Method	**Description**
getStoreType	Returns the type of archive.
isClosed	Returns a value that tells the user whether the archive is closed or not.
moveObject	Moves a document to the selected profile.
objectCount	Returns the number of documents in the archive.*
openObject	Opens a document in the archive.*
retrieveObject	Retrieves a document from the archive to a file on the local disk or to the data stream.
storeObject	Saves a document to the archive in the selected profile with the provided MIME type.

** IMiS®/Storage Connector .NET does not currently support these features.*

5.2.3 "Document"

The *Document* object represents an open document in the archive. It contains methods for saving, deleting and closing. Access to the server is enabled through the data stream. These objects also contain metadata about the document such as an identifier and the size of the document, whether the document was newly created, edited, saved, etc.

The *Document* object contains the following elements:

.NET	
Property	**Description**
AutoSave	Returns or sets a value that tells the user whether the document is automatically saved upon closing or not.
Created	Returns the date and time the document was created.*
DataStream	Returns the document's data stream.
DefaultExtension	Returns the default document extension.
DefaultMime	Returns the document's default MIME type.
Extensions	Returns a list of extensions for the document's MIME type.
Id	Returns the object identifier of the document.
IsClosed	Returns a value that tells the user whether the document is closed or not.
IsModified	Returns a value that tells the user whether the document has been modified or not.
IsNew	Returns a value that tells the user whether the document is newly created or not.
LastAccessed	Returns the date and time the document was last accessed.*
Mimes	Returns a list of MIME types for the document's extension.
Mode	Returns a value that specifies the mode in which the document is open.
Modified	Returns the date and time the document was last modified.*
Size	Returns the size of the document in bytes.*
Store	Returns the archive where the document is located.

Method	Description
Clone	Creates a copy or clone of the document.*
Close	Closes the document.
Delete	Deletes the document in the archive.
Move	Moves the document to the selected profile.*
Save	Saves the document.

** IMiS®/Storage Connector .NET does not currently support these features.*

Java	
Method	**Description**
clone	Creates a copy or clone of the document.*
close	Closes the document.
delete	Deletes the document in the archive.
getAccessMode	Returns a value that specifies the mode in which the document is open.
getAutoSave	Returns a value that tells the user whether the document is automatically saved upon closing or not.
getCreated	Returns the date and time the document was created.*
getDefaultExtension	Returns the document's default extension.
getDefaultMime	Returns the document's default MIME type.
getExtensions	Returns a list of extensions for the document's MIME type.
getId	Returns the object identifier of the document.
getInputDataStream	Returns the document's incoming data stream.
getLastAccessed	Returns the date and time the document was last accessed.*
getMimes	Returns a list of MIME types for the document's extension.
getModified	Returns the date and time the document was last modified.*
getOutputDataStream	Returns the document's outgoing data stream.
getSize	Returns the size of the document in bytes.
getStore	Returns the archive where the document is located.
isClosed	Returns a value that tells the user whether the document is closed or not.
isModified	Returns a value that tells the user whether the document has been modified or not.
isNew	Returns a value that tells the user whether the document is newly created or not.
move	Moves the document to the selected profile.*
save	Saves the document.
getAutoSave	Returns or sets a value that tells the user whether the document is automatically saved upon closing or not.

** IMiS®/Storage Connector Java does not currently support these features.*

5.2.4 "AuditLog"

The *AuditLog* object represents a message that can be used in the audit trail for certain operations with documents, including creating, opening, saving, moving and deleting.

The message must be delivered in the form of a C-style string.

The AuditLog object contains the following elements:

.NET	
Property	**Description**
Arguments	Returns or sets arguments for the audit trail message.
Message	Returns or sets the audit trail message.

Java	
Method	**Description**
getArguments	Returns arguments for the audit trail message.
getMessage	Returns the audit trail message.
setArguments	Sets arguments for the audit trail message.
setMessage	Sets the audit trail message.

5.3 Interface for IMiS®/ARChive Server 9

IMiS®/ARChive Server 9 introduces a new approach to archiving content using the so-called entity model.

The IMiS®/Storage Connector application program interface (API) for IMiS®/ARChive Server version 9 enables the user to easily connect to and work with archive servers.

It encompasses work with entities, metadata and content through registered archive users.

At present, only the .NET version of this API is available.

The interface for accessing IMiS®/ARChive Server version 9 is divided into three address spaces:

- *IMiS.StorageConnector* – contains enumerators, folders and interfaces, the most noteworthy of which are the *StorageConnector* class and the *IArchive* interface.
  The *StorageConnector* class represents the primary object of the IMiS®/Storage Connector interface.
  The *IArchive* interface represents an archive on the client side that is linked to a specific archive server.
  The *IMiS.StorageConnector* address space contains the following elements:

.NET	
Enumerator	**Description**
ArchiveCapabilities	Values representing the capabilities of the archive.
ArchiveType	Values representing the archive type.
AuditQueryObjectParamsScope	These values represent different types of restrictions on the range of parameters when searching the audit trail.
AuditQueryParamsType	These values represent different types of restrictions on groupings of parameters when searching the audit trail.
AuditQueryResultSortOrder	These values represent different types of sorting orders for audit trail search results.
AuditQuerySessionParamsScope	These values represent different types of restrictions on the range of session parameters when searching the audit trail.
AuthCryptoAlgorithm	These values represent the type of cryptographic algorithm used to set up a connection to the archive.
AuthCryptoCipherMode	These values represent the type of block cipher used when setting up a connection to the archive.
AuthCryptoKeySize	These values represent the size of the cryptographic key used when setting up a connection to the archive.
AuthType	These values represent the type of authentication with the archive.

Class	Description
AuditLog	Enables a message to be set for the audit trail.
AuditQuery	Enables parameters to be set for the audit trail.
StorageConnector	Enables archives to be opened and logging to be configured.
StorageConnectorException	Error running IMiS®/Storage Connector.

Interface	Description
IArchive	Defines operations on the archive.
IContentTypeResolver	Defines operations for getting an extension from the content type (MIME) and vice versa.

- *IMiS.StorageConnector.EntityModel* contains enumerators, classes and interfaces for the entity model used in IMiS®/ARChive Server version 9. *IClass*, *IFolder* and *IDocument* represent the different types of entities in the archive (class, folder and document).
  The *IEntityStub* interface represents publicly available metadata for an individual entity.
  The *IProperty* interface represents an individual piece of entity metadata.
  The *IContent* interface represents the content of a document in the form of a file available through the *IContentPart* interface.

  The *IMiS.StorageConnector.EntityModel* address space contains the following elements:

.NET	
Enumerator	**Description**
DeletionKind	These values represent a type of deletion.
EntityAccess	These values represent types of access to the archive.
EntityIdEncoding	These values represent various encodings of entity identifiers.
EntityIdKind	These values represent various kinds of entity identifiers.
EntityIdType	These values represent various types of entity identifiers.
EntityQueryScope	These values represent the various types of data about an entity the user would like to obtain.
EntityRightsFilter	These values represent filters for rights to entities.
EntitySortKeyDirection	These values represent the direction in which entities are sorted by keys.
EntityType	These values represent the entity types (class, folder or document).
EntityTypeFilter	These values represent various filters for entity types.
LogType	These values represent various system directories for exported, imported and transferred entities.
PropertyType	These values represent various property types for a piece metadata.
ReportType	These values represent various types of reports.
RetentionEntryFilter	These values represent various retention filters based on the type of entity.
RetentionEntryScope	These values represent various retention scopes based on the type of entity
SearchOptions	These values represent various options for searching.
SystemProperty	These values represent various system metadata.

Class	Description
ACLFilterItem	A filter for user rights.
EntityFilter	Enables filtration parameters to be set.
EntityFilterItem	Represents the basis for individual filters.
EntitySortKey	Represents a key for sorting entity collections.

Interface	Description
IACL	Represents the ACL (Access Control List), a collection of user rights for a specific entity.
IBinaryValue	Represents binary content for metadata.
IClass	Represents a class in the classification scheme in the archive.
IContent	Enables the reading and editing of entity file content.
IContentPart	Represents the content of an entity in the form of a file.
IDeletionStub	Represents the metadata of a deleted entity.
IDispositionHold	Represents the definition of disposition hold.
IDispositionHoldEntry	Represents an entry for entity disposition hold.
IDocument	Represents a document in a class or folder in the archive.
IEmailEntity	Enables access to email metadata.
IEntity	Represents an entity in the classification scheme in the archive.
IEntityACLEntry	Represents user rights for an entity.
IEntityRights	Enables the viewing and editing of user rights for an entity.
IEntityStub	Represents publicly available metadata about an entity.
IFolder	Represents a folder in a class or folder in the archive.
IMoveDetails	Enables access to the metadata of a moved entity.
IPhysicalEntity	Enables access to physical entity metadata.
IPickListValue	Represents a value for metadata with predefined valid values.
IProperty	Enables the reading and editing of metadata.
IPropertyACLEntry	Represents user rights for metadata.
IPropertyRights	Enables the viewing and editing of user rights for metadata.
IRDSEntry	Represents user-related data about retention and disposition.
IRDSList	Represents a data collection about retention and disposition for an entity.
IReadOnlyContent	Enables the reading of entity file content.
IReadOnlyEntityRights	Enables the viewing of user rights for an entity.
IReadOnlyProperty	Represents read-only metadata.
IReadOnlyPropertyRights	Enables the viewing of user rights for metadata.
IReadOnlyRetentionPolicyContext	Represents the entry context for read only entity retention policy.
IRetention	Represents entry collections for retention policy and entity disposition hold.
IRetentionPolicy	Represents the definition of retention policy.
IRetentionPolicyContext	Represents retention policy context for entity retention policy.
IRetentionPolicyEntry	Represents the entry for entity retention policy.
IRetentionPolicySnapshot	Represents a retention policy setting at the exact time of review preparation of disposition actions.
IReview	Represents the review process of disposition actions.
IReviewStub	Represents publicly available metadata for review process of disposition actions.
ISearchedEntityStub	Represents metadata of a searched entity.
ISecurityClassChangeDetails	Enables access to data on changes made to the security class of an entity.
IStringMaxValue	Represents text content for metadata.
ITemplate	Represents a template for entity creation.
ITransferDetails	Enables access to the metadata of a transferred entity.

- *IMiS.StorageConnector.Services* contain enumerators, classes and interfaces for services enabled by IMiS®/ARChive Server version 9.

  The *IDirectory* interface makes it possible to obtain information on registered archive users.

  The *IDirectoryEntity* represents an archive "user". The user can be a group or an individual user.

  The *IMiS.StorageConnector.Services* address space contains the following elements:

.NET	
Enumerator	**Description**
DirectoryEntityType	Values that represent the archive user type (user group or individual user).

Class	**Description**
Discovery	Enables searching in archives above the archive server.

Interface	**Description**
IArchiveDescriptor	Represents a description of the archive above the archive server.
IDirectory	Makes it possible to obtain information on registered archive users.
IDirectoryEntity	Represents a registered archive user.

A detailed description of the IMiS®/ARChive Server version 9 components outlined above is presented below.

5.3.1 "StorageConnector"

StorageConnector is the primary object of the IMiS®/Storage Connector application program interface (API). Access to the singleton object instance is managed through the *Instance* property, which is only valid until the *FreeInstance* method is called up. This object contains the *OpenArchive* method for opening the archive on the client side (*see chapter 5.3.2 "IArchive"*) for different archive servers. At the time of writing, only IMiS®/ARChive Server is supported. This object also supports logging settings and contains constants representing the names of different options which are delivered together with the suitable values when opening an archive.

Below the archive to IMiS®/ARChive Server is marked with the prefix *IMiSARChive*, which is equivalent to the *ArchiveType* value in the interface.

The *StorageConnector* object contains the following elements:

.NET	
Constant	**Description**
MAX_OPTION_NAME_LENGTH	The maximum length of names for the audit trail.
OptionApplicationName	Option for the *IMiSARChive* archive that determines the name of the application for the audit trail.
OptionAuthCryptoAlgorithm	Option for the *IMiSARChive* archive that determines the type of cryptographic algorithm used in authentication.
OptionAuthCryptoCipherMode	Option for the *IMiSARChive* archive that determines the type of operation with block ciphers in authentication.
OptionAuthCryptoKeySize	Option for the *IMiSARChive* archive that determines the size of the authentication key.
OptionAuthKey	Option for the *IMiSARChive* archive object that determines the authentication key.
OptionAuthType	Option for the *IMiSARChive* archive that determines the type of authentication.
OptionComputerName	Option for the *IMiSARChive* archive that determines the name of the computer for the audit trail.
OptionDiscoveryArchiveTypes	Option for determining the types of archives returned when searching archives.
OptionEntityCollectionPageCount	Option for the *IMiSARChive* archive that determines the number of pages in entity collections.
OptionEntityCollectionPageSize	Option for the *IMiSARChive* archive that determines the number of entities per page in entity collections.
OptionEntityIdEncoding	Option for the *IMiSARChive* archive that determines how an object identifier is encoded.
OptionEntityIdType	Option for the *IMiSARChive* archive that specifies what kind of (short/long) object identifiers the program is dealing with.
OptionLocalAddress	Option for the *IMiSARChive* archive that determines the local IP address of the user for the audit trail.
OptionMaxSessionsPerUser	Option for the *IMiSARChive* archive that determines the maximum number of sessions per user.
OptionUserName	Option for the *IMiSARChive* archive object that determines the name of the user for authentication with UserCredentials and/or for the audit trail.
OptionUserPassword	Option for the *IMiSARChive* archive object that determines the user password for authentication with *UserCredentials*.

.NET	
Property	**Description**
ContentTypeResolver	Returns the predefined interface for getting an extension from the content type (MIME) and vice versa.
CustomContentTypeResolver	Returns or sets a custom interface for getting an extension from the content type (MIME) and vice versa.
Instance	Returns a singleton instance of the *StorageConnector* object. The first time the program is used an instance is created. This instance is returned until the *FreeInstance* method is called up, at which point the singleton instance is no longer valid.
LogHandlers	Returns a list of external logging handlers.
LogInternal	Returns or sets a value that tells the user whether internal logging is performed or not.
LogLevel	Returns or sets the current level of logging.
LogLocale	Returns or sets regional settings for logging.
VersionInfo	Returns the IMiS®/StorageConnector version.

Method	Description
FreeInstance	Terminates the singleton instance of the *StorageConnector* object. Once this method is used, calling up the *Instance* property will not return a valid instance, as this instance no longer exists.
LogAddHandler	Adds a logging handler to the list of external logging handlers.
LogClearHandlers	Clears the list of external logging handlers.
LogRemoveHandler	Removes the logging handler from the list of external logging handlers.
OpenArchive	Opens the archive for IMiS®/ARChive Server (the *IMiSARC* archive) through the provided network address (host name or IP address) and port.
ContentTypeResolver	Makes it possible to get an extension from the content type (MIME) and vice versa.

5.3.2 "IArchive"

The *IArchive* interface represents an archive on the client side that is linked to an archive server.

It contains the following operations: creating and opening entities (classes, folders and documents), getting the audit trail through the *AuditLogQuery* method and archive search using the *Search* method.

A message for the audit trail report can also be set for certain operations using the *AuditLog* object (*see chapter 5.3.8 "Auditlog"*). Note that a message must be separately set for each operation.

The *IArchive* interface has the following elements:

.NET	
Property	**Description**
AuditLog	Returns the *AuditLog* object for setting a message for the audit trail.
Capabilities	Returns archive capabilities for the current user.
Directory	Returns the *IDirectory* object for identifying archive users.
DispositionHolds	Returns the collection of all disposition holds on the archive.
EffectiveRights	Returns the effective rights of the user on the root archive.
EntityCollectionPageCount	Returns or sets the number of pages in an entity collection.
EntityCollectionPageSize	Returns or sets the page size in an entity collection.
IsClosed	Returns a value that tells the user whether the archive is closed or not.
RetentionPolicies	Returns the collection of all retention policies on the archive.
Template	Returns a collection of all entity templates in the archive
User	Returns the user name of the user currently logged in.

Method	Description
AuditLogQuery	Performs an action that gets the audit trail.
Close	Closes the archive.
CreateClass	Creates a class on the root archive or in the selected class.
CreateDocument	Creates a document in the selected class or folder.
CreateFolder	Creates a folder in the selected class or folder.
CreateReview	Creates a review of disposition actions.
DeleteEntity	Deletes an entity listed with an entity identifier from the archive.
GetDeletedEntities	Returns a collection of deleted entities.
GetEntityInfo	Returns public data about one or more entities listed with their identifiers.
GetLogEntries	Returns a collection of entities from the selected system directory.
GetReport	Returns the selected type of report.
GetReviewInfo	Returns public data on review of disposition actions.
GetReviews	Returns a review of disposition actions.
GetRootClasses	Returns a collection of classes on the root archive.
GetSytemPropertyName	Returns the name of a system property.
MoveEntity	Moves an entity to the selected parent entity.
OpenClass	Opens a class in the archive.
OpenDocument	Opens a document in the archive.
OpenEntity	Opens an entity in the archive.
OpenFolder	Opens a folder in the archive.
OpenReview	Opens a review of disposition actions.
Search	Performs a search of the archive.
SetEntitySecurityClass	Sets the security class of an entity.
SetEntityStatus	Sets the status of an entity.

5.3.3 "IDirectory" and "IDirectoryEntity"

The *IDirectory* interface represents a service for identifying registered archive users. Registered users are presented through the *IDirectoryEntity* interface, which is uniquely defined through the *Subject* property. This service is available for authenticated archive users.

The *IDirectory* interface contains the following elements:

.NET	
Property	**Description**
InvalidDirectoryEntity	Returns an invalid user.
Members	Returns all registered archive users.
Parent	Returns a reference to the archive.

Method	Description
ChangePassword	Enables a user password to be changed.
GetGroupMembers	Returns all registered archive users for the selected group.

The *IDirectoryEntity* interface contains the following elements:

.NET	
Property	**Description**
Description	Returns a description of the user.
Email	Returns a user's email address.
FirstName	Returns a user's first name.
LastName	Returns a user's last name.
SecurityClass	Returns a user's assigned security class.
Subject	Returns a unique user identifier.
Type	Returns the type of user (user group or individual user).

5.3.4 "IEntityStub"

The *IEntityStub* interface represents publicly available data on an individual entity. These data include the title (the *Title* property), the classification code (the *ClassificationCode* property), a collection of public metadata obtained through the *Properties* property, and methods such as *Open* for opening an entity in read-only or read and write mode and *Search* for searching subentities.

The *IEntityStub* interface contains the following elements:

.NET	
Property	**Description**
Accessed	Returns the date and time the entity was last accessed.
Archive	Returns a reference to the archive.
ClassificationCode	Returns the classification code in canonicalized form.
Closed	Returns the date and time the entity status was changed to Closed.
Contents	Returns the entity's content collection.
Created	Returns the date and time the entity was created.
Creator	Returns the author of the entity.
Description	Returns a description of the entity.
EffectiveRights	Returns the effective rights of the current user on the entity.
ExternalIds	Returns a list of external identifiers of the entity.
Id	Returns the entity's internal identifier.
IsChildClassificationCodeGenerated	Returns a value that tells the user whether classification codes are automatically generated for child or subentities or not.
Keywords	Returns a collection of keywords for the entity.
Modified	Returns the date and time the entity was last modified.
Opened	Returns the date and time the entity's status was changed to Opened.
Owner	Returns the owner of the entity.
Parent	Returns the parent entity.
Properties	Returns a collection of metadata.
PublicClassificationCode	Returns the classification code in public form.
Retention	Returns the collection of retention policies and disposition holds.

.NET (continued)	
Property	**Description**
SecurityClass	Returns the security class of the entity. The following values are possible: - Unclassified: there are no special restrictions on accessing the entity. - Restricted: the entity is internal in nature. Only users with a Restricted security class or higher may access the entity. - Confidential: the entity is confidential in nature. Only users with a Confidential security class or higher may access the entity. - Secret: the entity is secret in nature. Only users with a Secret security class or higher may access the entity. - Top Secret: the entity is top secret. Only users with a Top Secret security class or higher may access the entity.
Significance	Returns the significance of the entity. The following values are possible: - Vital: the entity is of vital importance. - Permanent: the entity is permanent. - Retain: the entity is marked for retention. - Delete: the entity is marked for deletion.
Status	Returns the status of the entity (of a class or a folder or of a document located directly below a class). The following values are possible: - Opened: a user may edit the entity provided they have adequate rights (write rights). - Closed: users can no longer edit the entity.
SubEntityCount	Returns the absolute number of subentities.
SubEntityTemplates	Returns a collection of templates for creating subentities.
Template	Returns the template used to create the entity.
Title	Returns the title of the entity.
Type	Returns the type of entity.

Method	**Description**
AreValuesInhereted	Returns a value that tells the user whether an entity's system properties are inherited or not.
Delete	Deletes the entity.
GetReport	Returns the selected type of report.
GetSubEntities	Returns a collection of child subentities in line with the selected filters and sorting keys.
Move	Moves the entity below the selected parent entity.
Open	Opens the entity in the selected mode (read only or read and write).
Search	Performs a search below the entity.
SetSecurityClass	Sets the security class of the entity.
SetStatus	Sets the status of the entity.

5.3.5 "IEntity", "IClass", "IFolder" and "IDocument"

The *IEntity* interface represents an entity that was opened in read only or read and write mode. It contains the shared properties and methods of entities in the entity model. The *IClass*, *IFolder* and *IDocument* interfaces represent specialized entities for class, folder and document.

Besides publicly accessible data on the entity (*see chapter 5.3.4 "IEntityStub"*) the following properties are also available: *EffectiveRights* presents an overview of the currently logged in user's rights for an entity, *PhysicalEntity* brings together metadata about a physical entity, and the *Save* method is used to save changes to an entity.

The *IEntity* interface contains the following elements:

.NET	
Property	**Description**
Accessed	Returns the date and time the entity was last accessed.
ACL	Returns a collection of user rights for the specified entity.
ArchivalInformationalPackage	Returns the archival information package in base64 format.
Archive	Returns a reference to the archive.
AuditLog	Returns the *AuditLog* object for setting a message for the audit trail.
ClassificationCode	Returns or sets a classification code in canonicalized form.
Closed	Returns the date and time the entity's status was changed to Closed.
Created	Returns the date and time the entity was created in the archive.
Creator	Returns the author of the entity.
Description	Returns or sets a description of the entity.
EffectiveRights	Returns the effective rights of the current user for the entity.
EmailEntity	Returns metadata about email.
EvidenceRecord	Returns an evidence record in base64 format.
ExternalIds	Returns or defines a list of external identifiers of the entity.
Id	Returns the entity's internal identifier.
IsClosed	Returns a value that tells the user whether the entity is closed or not.
Keywords	Returns or defines keywords for the entity.
Mode	Returns a value that specifies the mode in which the entity is open.
Modified	Returns the date and time the entity was last modified.
MoveDetails	Returns a collection of metadata of the moved entity.
Opened	Returns the date and time the entity's status was changed to Opened.
Owner	Returns or sets the owner of the entity.
Parent	Returns the parent entity.
PhysicalEntity	Returns metadata about physical records.
Properties	Returns a collection of metadata.
PublicClassificationCode	Returns the classification code in public format.
SaveLog	Returns a record generated when the entity is saved.
SecurityClass	Returns or sets the security class of the entity. The following values are possible: - Unclassified: there are no special restrictions on accessing the entity. - Restricted: the entity is internal in nature. Only users with a Restricted security class or higher may access the entity. - Confidential: the entity is confidential in nature. Only users with a Confidential security class or higher may access the entity. - Secret: the entity is secret in nature. Only users with a Secret security class or higher may access the entity. - Top Secret: the entity is top secret. Only users with a Top Secret security class or higher may access the entity.

.NET (continued)	
Property	**Description**
SecurityClassChangeDetails	Returns a list of changes to the entity's security class.
Significance	Returns or sets the significance of the entity. The following values are possible: - Vital: the entity is of vital importance. - Permanent: the entity is permanent. - Retain: the entity is marked for retention. - Delete: the entity is marked for deletion.
Status	Returns the status of the entity (of a class or a folder or of a document located directly below a class) The following values are possible: - Opened: a user may edit the entity provided they have adequate rights (write rights). - Closed: users can no longer edit the entity.

.NET	
Property	**Description**
SubEntityCount	Returns the absolute number of child or subentities.
SubEntityTemplates	Returns a collection of templates for creating subentities.
Template	Returns the template used to create the entity.
Title	Returns or sets the title of the entity.
TransferDetails	Returns metadata about a transferred entity.
Type	Returns a value that specifies the entity type.

Method	Description
AreValuesInhereted	Returns a value that tells the user whether an entity's system properties are inherited or not.
Close	Closes an entity.
CreateBinaryValue	Creates binary content in the archive.
CreateStringMaxValue	Creates text or string content in the archive.
GetPickListValues	Returns a list of the entity's predefined system properties values.
GetReport	Returns the selected type of report.
GetSubEntities	Returns a collection of subentities in line with the selected filters and sorting keys.
Save	Saves changes to the entity.
Search	Performs a search below the entity.

Besides the elements of the *IEntity* interface, the *IClass* interface also contains the following elements:

.NET	
Property	**Description**
IsChildClassificationCodeGenerated	Returns or sets a value that tells the user whether classification codes are automatically generated for child or subentities or not.

Method	Description
CreateClass	Creates a subclass below the class.
CreateDocument	Creates a document below the class.
CreateFolder	Creates a folder below the class.

Besides the elements of the *IEntity* interface, the *IFolder* interface also contains the following elements:

.NET	
Method	**Description**
CreateDocument	Creates a document below the folder.
CreateFolder	Creates a subfolder below the folder.

Besides the elements of the *IEntity* interface, the *IDocument* interface also contains the following elements:

.NET	
Property	**Description**
Content	Returns the entity's content collection.
CustomContents	Returns a custom collection of the entity's content.

Method	Description
CreateContentPart	Creates a file in the archive.

5.3.6 "IReadOnlyProperty" and "IProperty"

The *IReadOnlyProperty* interface represents metadata intended only for reading or viewing. It contains properties that describe metadata and metadata values. The *Type* property returns the metadata type, and the *ValueCount* property returns the number of metadata values. Access to the metadata values is enabled with the *GetValue* and *GetValues* methods.
The *IProperty* executable interface represents metadata that can be edited. Besides the values and methods of the *IReadOnlyProperty* interface,
it also contains the *EffectiveRights* property for viewing the effective rights of the current user for the metadata and the *SetValue* and *SetValues* methods for setting one or more metadata values.
The *IReadOnlyProperty* interface contains the following elements:

.NET	
Property	**Description**
AreValuesInhereted	Returns a value that tells the user whether metadata values are inherited or not.
Description	Returns a description of the metadata.
IsAppendOnly	Returns a value that tells the user whether metadata values can only be appended (added) or not.
IsIncludedInAIP	Returns a value that tells the user whether metadata values form a part of the archival information package (AIP).
IsInherited	Returns a value that tells the user whether a metadata value is inherited.
IsMultiValue	Returns a value that tells the user whether metadata can have more than one value.
IsPickList	Returns a value that tells the user whether the metadata has predefined valid values.
IsPublic	Returns a value that tells the user whether the metadata is publicly accessible.
IsReadOnly	Returns a value that tells the user whether the metadata is read only.

.NET (continued)	
Property	**Description**
IsRequired	Returns a value that tells the user whether the metadata is required.
IsSearchable	Returns a value that tells the user whether the metadata's values can be used to search.
IsUnique	Returns a value that tells the user whether the metadata value must be unique.
Name	Returns the unique name of the metadata.
Owner	Returns the *IEntityStub* from the entity linked to the metadata.
PickListValues	Returns a list of predefined metadata values.
Type	Returns the type of metadata.
ValueCount	Returns the number or count of metadata values.

Method	Description
GetValue	Returns the metadata value.
GetValues	Returns a list of metadata values.
GetXmlValue	Returns the metadata value in XML-equivalent format.
GetXmlValues	Returns a list of metadata values in XML-equivalent format.

Besides the elements of the *IReadOnlyProperty* interface, the *IProperty* interface also contains the following elements:

.NET	
Property	**Description**
CommittedValueCount	Returns the number or count of saved metadata values.
EffectiveRights	Returns the effective rights of the current user for the metadata.
Owner	Returns the *IEntity* from the entity linked to the metadata.

Method	Description
Clear	Clears the metadata value(s).
SetValue	Sets the metadata value.
SetValues	Sets a list of metadata values.
SetXmlValue	Sets the metadata value in XML-equivalent format.
SetXmlValues	Sets a list of metadata values in XML-equivalent format.

5.3.7 "IReadOnlyContent", "IContent" and "IContentPart"

The *IReadOnlyContent* interface represents the content of an entity intended only for reading or viewing. The *IContent* executable interface represents the content of an entity that may be edited.

The content of an entity consists of one or more files accessed through the *IContentPart* interface.

The *IReadOnlyContent* interface contains the following elements:

.NET	
Property	**Description**
Name	Returns the unique name of the content collection.
Owner	Returns the *IEntityStub* from the entity linked to the content collection.
PartsCount	Returns the number or count of contents in the content collection.

Method	Description
GetParts	Returns a list of entity contents.

Besides the elements of the *IReadOnlyContent* interface, the *IContent* interface also contains the following elements:

.NET	
Property	**Description**
EffectiveRights	Returns the effective rights of the current user for the metadata.
Owner	Returns the *IEntity* from the entity linked to the content collection.

Method	Description
Clear	Clears an entity's content collection.
SetParts	Sets a list of entity contents.

The *IContentPart* interface contains the following elements:

.NET	
Property	**Description**
Accessed	Returns the date and time the file was last accessed.
ContentTypes	Returns a list of content types (MIME) for the file.
Created	Returns the date and time the file was created.
DefaultContentType	Returns the default content type (MIME) for the file.
DefaultExtension	Returns the default file extension for the file.
Description	Returns or sets a description of the file.
Extensions	Returns a list of extensions for the file.
Id	Returns the unique identifier of the content.
Modified	Returns the date and time the file was last modified.
Size	Returns the number of content values.

Method	Description
OpenDataStream	Opens the datastream of the file in the selected mode (read only or read and write).

5.3.8 "IRetention", "IRetentionPolicyEntry", "IRetentionPolicyContext" and "IDispositionHoldEntry"

The *IRetention* interface represents entry collections for retention policy and entity disposition hold. It enables access to the collections and saving changes. The retention policy entry of an entity is represented by the *IRetentionPolicyEntry interface* that is specified in detail by the *IRetentionPolicyContext* interface. The entity disposition hold entry is represented by the *IDispositionHoldEntry* interface.

The *IRetention* interface contains the following elements:

.NET	
Property	**Description**
DispositionHoldEntries	Returns the collection of entries for entity disposition hold.
Owner	Returns the *IEntityStub* from the entity that is connected to the collection of retention policy entries and entity disposition hold.
PolicyEntries	Returns the collection of entries for entity retention policies.

Method	**Description**
AddDispositionHold	Adds an entry of entity disposition hold.
AddPolicyEntry	Adds an entry of entity retention policies.
Commit	Saves the changes of retention policy collections and entity disposition holds.
Revert	Rejects changes in retention policy collections and entity disposition holds.

The *IRetentionPolicyEntry* interface contains the following elements:

.NET	
Property	**Description**
Definition	Returns the definition of entity retention policy.
ExplicitContext	Returns the explicit context entry for entity retention policy.
InheritedContext	Returns the inherited context entry for entity retention policy.

Method	**Description**
Delete	Deletes the explicit context entry for entity retention policy.

The *IRetentionPolicyContext* interface contains the following elements:

.NET	
Property	**Description**
Filter	Returns or sets an entity retention policy context filter.
Owner	Returns the *IRetentionPolicyEntry* that is connected to the context.
Scope	Returns the scope of context entity retention policy.

The *IDispositionHoldEntry* interface contains the following elements:

.NET	
Property	**Description**
Definition	Returns the definition of entity disposition hold.
IsInherited	Returns the value that tells the user whether the entry for entity disposition hold is inherited.

Method	Description
Delete	Deletes an explicit entry for entity disposition hold.

5.3.9 "IReviewStub"

The *IReviewStub* interface represents publicly available review metadata. The interface is a limited version of *IEntityStub* (*see chapter 5.3.4 »IEntityStub«*) interface with two additions:

- the *State* property that represents the review state;
- The *Message* property, where an archive server logs operation messages during the review.

The *IReviewStub* interface contains the following elements:

.NET	
Property	**Description**
Accessed	Returns the date and time the review was last accessed.
Archive	Returns the reference to the archive.
ClassificationCode	Returns the classification code in canonicalized form.
Closed	Returns the date and time the reviewing status was changed to Closed.
Content	Returns the content collection review.
Created	Returns the date and time the review was created.
Creator	Returns the author of review.
Description	Returns a description of the review.
EffectiveRights	Returns the effective rights of the current user on the review.
Id	Returns the review's internal identifier.
Keywords	Returns a collection of keywords for the review.
Message	Returns the message from the archive server that represents the state or error.
Modified	Returns the date and time the review was last modified.
Opened	Returns the date and time the review's status was changed to Opened.
Owner	Returns the owner of the review.
PublicClassificationCode	Returns the classification code in public form.

.NET (continued)	
Property	**Description**
State	Returns the review's state. The following values are possible: - Unknown: the review's state is unknown; - Created: a review is created; - Preparing: a review is being prepared; - InReview: the review is in review; - Completing: the review is in the process of completion; - Discarded: the review has been discarded; - Failed: the review has failed.
Status	Returns the status of the review. The following values are possible: - Opened: the review can be edited by a user with adequate rights (write rights). - Closed: users can no longer edit the review.
SubEntityTemplates	Returns a collection of templates for creating subentities.
Template	Returns the template used to create the review.
Title	Returns the title of the review.

Method	Description
AreValuesInhereted	Returns a value that tells the user whether the review's system properties are inherited or not.
Open	Opens the review in the selected mode (read only or read and write).
SetStatus	Sets the status of the review.

5.3.10 "IReview"

The *IReview* interface represents a review that is opened in read only or read and write mode. The interface represents a limited version of the IEntity interface *(see chapter 5.3.5 »IEntity«, »IClass«, »IFolder« in »IDocument«)*.

Besides publicly availabe review data *(see chapter 5.3.9 »IReviewStub«)*, the following properties are also available:

- *Action* for selecting an action in the review.
- *Members* for selecting members in the review..
- ReviewedItems and ScheduledItems with a collection of entities that are reviewed or subjects to review.

The IReview interface contains the following elements:

.NET	
Property	**Description**
Accessed	Returns the date and time the review was last accessed.
ACL	Returns a collection of user rights for review.
Action	Returns the review action. The following values are possible: - Reviewing: a review is in progress - Completed: a review is completed - Discard: a review will be discarded.
ArchivalInformationalPackage	Returns the archival information package (AIP) in base64 format.
Archive	Returns a reference to the archive.
AuditLog	Returns the AuditLog object for setting a meesage for the audit trail.
ClassificationCode	Returns or sets a classification code in canonicalized form.
Closed	Returns the date and time the review's status was changed to Closed.
Created	Returns the date and time the review was created in the archive.
Comments	Returns comments on the review.
Creator	Returns the author of the review.
Description	Returns or sets a description of the review.
Documents	Returns a collection of review documents.
EffectiveRights	Returns the effective rights of the current user for the review.
EvidenceRecord	Returns an evidence record in base64 format.
Id	Returns the entity's internal identifier.
IsClosed	Returns a value that tells the user whether the review is closed or not.
Keywords	Returns or sets a collection of keywords for the review.
Members	Returns the members of the commision that have implemented the review.
Message	Returns a message from the archive server that represents the state or error in the review process.
Mode	Returns a value that specifies the mode in which the review is open.
Modified	Returns the date and time the review was last modified.
Opened	Returns the date and time the review's status was changed to Opened.
Owner	Returns or sets the owner of the entity.
PublicClassificationCode	Returns the classification code in public format.
QueryExpression	Returns a query expression used when creating the review.
RetentionPolicies	Returns a collection of retention policies used when creating the review.
ReviewedItems	Returns a collection of entities that have been reviewed.
SaveLog	Returns a record generated when the entity is saved.
ScheduledItems	Returns a collection of entities that are subjects to review.
Scope	Returns an entity that defines the scope of the review.

.NET (continued)	
Property	**Description**
Status	Returns or sets the status. The following values are possible: - Opened: a user may edit the review if he/she has appropriate rights (write rights). - Closed: a user can no longer edit the review.
State	Returns the review state. The following values are possible: - Unknown: the review is unknown. - Created: a review is created. - Preparing: a review is being prepared. - InReview: the review is in review. - Completing: the review is in the process of completion. - Completed: the review is completed. - Discarded: the review has been discarded. - Failed: the review has failed.
SubEntityCount	Returns the absolute number of subentities.
SubEntityTemplates	Returns a collection of templates for creating subentities.
Template	Returns the template used to create the review.
Title	Returns or sets the title of the review.

Method	Description
AreValuesInhereted	Returns a value that tells the user whether a review's system properties are inherited or not.
Close	Closes the review.
CreateDocument	Creates a document in the review.
GetPickListValues	Returns a list of the review's predefined system properties values.
Save	Saves changes to the review.

5.3.11 "AuditLog"

The *AuditLog* object represents a message that can be used in the audit trail for certain operations with an entity, including creating, opening, saving, moving and deleting.

The message must be delivered in the form of a C-style string.

The *AuditLog* object contains the following elements:

.NET	
Property	**Description**
Arguments	Returns or sets arguments for the audit trail message.
Message	Returns or sets the audit trail message.

5.3.12 "AuditQuery"

The *Auditquery* object represents parameters for getting the audit trail. The parameters are divided into three groups: parameters linked to the session (network address, computer name and user name), parameters linked to objects (object identifiers and action identifiers) and dates.

Session and object parameters can be delivered as a list or a series. The series determines the first and last values. Besides these parameters, there is also a parameter that determines the order in which the audit trail is sorted. The audit trail is returned in the form of a data stream through the *AuditLogQuery* method of the *IArchive* interface *(see chapter 5.3.2 "IArchive")*.

The *AuditLog* object contains the following elements:

.NET	
Property	**Description**
Addresses	Returns a list of network addresses that form a part of the session parameters.
AddressFrom	Returns or sets the first network address in a series of network addresses that form a part of the session parameters.
AddressTo	Returns or sets the last network address in a series of network addresses that form a part of the session parameters.
ComputerNameFrom	Returns or sets the first computer name in a series of computer names that form a part of the session parameters.
ComputerNames	Returns a list of computer names that form a part of the session parameters.
ComputerNameTo	Returns or sets the last computer name in a series of computer names that form a part of the session parameters.
DateFrom	Returns or sets the first date in a series of dates that form a part of the date parameters.
DateTo	Returns or sets the last date in a series of dates that form a part of the date parameters.
EventIdFrom	Returns or sets the first event identifier in a series of event identifiers that form a part of the object parameters.
EventIdQueryType	Returns or sets a value that tells the user whether event identifiers are arranged as a list or a series.
EventIds	Returns a list of event identifiers that form a part of the object parameters.
EventIdTo	Returns or sets the last event identifier in a series of event identifiers that form a part of the object parameters.
ObjectIds	Returns a list of object identifiers that form a part of the object parameters.
ObjectScope	Returns or sets a value that tells the user whether the object parameters are object identifiers, event identifiers or both.
SessionQueryType	Returns or sets a value that tells the user whether session parameters are arranged as a list or a series.
SessionScope	Returns or sets a value that tells the user whether session parameters are network addresses, user names, computer names or all of these.
SortOrder	Returns or sets a value that tells the user the sorting order of the audit trail parameters (session, object and date parameters).
UserNameFrom	Returns or sets the first user name in a series of user names that form a part of the session parameters.

.NET (continued)	
Property	**Description**
UserNames	Returns a list of user names that form a part of the session parameters.
UserNameTo	Returns or sets the last user name in a series of user names that form a part of the session parameters.

5.4 Examples of use

The key feature of the IMiS®/Storage Connector interface is the transfer of objects between the application server and IMiS®/ARChive Server. Here transfer means saving objects on the archive server and taking objects from the archive server.

5.4.1 Initializing IMiS®/Storage Connector

Running IMiS®/Storage Connector starts by initializing the StorageConnector instance.
This occurs the first time the *StorageConnector* class is used.

Getting an instance of the *StorageConnector* object follows these steps:

.NET
StorageConnector sc = StorageConnector.Instance;

Java
StorageConnector sc = StorageConnector.getInstance();

The instance is set and valid until finalization, which is described in the following chapter (*see chapter 5.4.2 Finalizing IMiS®/Storage Connector*). After finalization, the steps described above will return an undefined value.

5.4.2 Finalizing IMiS®/Storage Connector

Finalizing sees to the proper shutting down of the IMiS®/Storage Connector interface.
It is performed when the instance of the *StorageConnector* object is freed. This is done when the instance of the *StorageConnector* object is no longer used.

Freeing an instance of the StorageConnector object follows these steps:

.NET
StorageConnector.FreeInstance();

Java
StorageConnector.freeInstance();

With this method, the StorageConnector instance longer exists. Obtaining an instance of the StorageConnector object as described in the previous section *(see chapter 5.4.1 Initializing IMiS®/Storage Connector)* will return an undefined value.

5.4.3 Examples for IMiS®/ARChive Server version 7

This chapter will present some of the most frequent examples of ways IMiS®/Storage Connector is used with IMiS®/ARChive Server version 7. These are operations for opening the archive, saving and retrieving documents and providing data for the audit trail.

5.4.3.1 Opening the archive

To open the archive, the following are needed: a *StorageConnector* instance *(see chapter 5.4.1 Initializing IMiS®/Storage Connector)*, the network address of the server (host name or IP address) and a suitable port.

To open the archive with PreSharedKey authentication, a collection of additional option parameters must be created and sent to the method for opening the archive.

An example of opening the archive without authentication:

.NET

```
StorageConnector sc = IMIS_STORAGE_CONNECTOR;
string host = "iarc.acme.com";
int port = 16807;
Storage stg = sc.OpenIMiSARCStorage(host, port);
```

Java

```
StorageConnector sc = IMIS_STORAGE_CONNECTOR;
String host = "iarc.acme.com";
int port = 16807;
Storage stg = sc.openIMiSARCStorage(host, port);
```


An example of opening the archive with *PresharedKey* authentication:

.NET

```
StorageConnector sc = IMIS_STORAGE_CONNECTOR;
string host = "iarc.acme.com";
int port = 16807;
IDictionary options = new SortedList();

options.Add(StorageConnector.OptionAuthType, AuthType.PreSharedKey);
options.Add(StorageConnector.OptionAuthKey, "psk1");

Storage stg = sc.OpenIMiSARCStorage(host, port, options);
```

Java

```
StorageConnector sc = IMIS_STORAGE_CONNECTOR;
String host = "iarc.acme.com";
int port = 16807;
Map options = new TreeMap();

options.put(StorageConnector.OPTION_AUTH_TYPE, new Integer(AuthType.PRE_SHARED_KEY));
options.put(StorageConnector.OPTION_AUTH_KEY, "psk1");

Storage stg = sc.openIMiSARCStorage(host, port, options);
```

5.4.3.2 Saving objects

Saving objects to the archive through the IMiS®/Storage Connector interface takes place in the following steps:

- Obtaining a *StorageConnector* instance *(see chapter 5.4.1 Initializing IMiS®/Storage Connector)*.
- Opening the archive with a network address (host name or IP address) and a port *(see chapter 5.4.4.1 Opening the archive)*.
- Saving the document to the archive as an object on the archive server.

.NET

```
Storage stg = IMIS_ARCHIVE_V7;
string fileName = "c:\acme.tif";
string profile = "Documents";

string objectId = stg.StoreObject(fileName, profile);
```


Java

```
Storage stg = IMIS_ARCHIVE_V7;
String fileName = "c:\acme.tif";
String profile = "Documents";

String objectId = stg.storeObject(fileName, profile);
```

5.4.3.3 Retrieving objects

Retrieving objects saved on the archive through the IMiS®/Storage Connector interface takes place in the following steps:

- Obtaining a *StorageConnector* instance *(see chapter 5.4.1 Initializing IMiS®/Storage Connector)*.
- Opening the archive with a network address (host name or IP address) and port. *(see chapter 5.4.4.1 Opening the archive)*.
- Retrieving a document listed with an object identifier from the archive server.

.NET

```
Storage stg = IMIS_ARCHIVE_V7;
string objectId = "e1aeed50688b8fd6df2b1aa93a8bd08620b7332561d84016b80428b69fe45e49";

string fileName = stg.RetrieveObject(objectId);
```

Java

```
Storage stg = IMIS_ARCHIVE_V7;
String objectId = "e1aeed50688b8fd6df2b1aa93a8bd08620b7332561d84016b80428b69fe45e49";

String fileName = stg.retrieveObject(objectId);
```

5.4.3.4 Providing data for the audit trail

The audit trail is a feature of IMiS®/ARChive Server. It is a log that records operations performed on objects. If this feature is enabled on the server, clients are required to provide audit information when opening the archive. This information must be provided through a collection of option parameters.

An example of opening the archive with an audit trail:

.NET

```
StorageConnector sc = IMIS_STORAGE_CONNECTOR;
string host = "iarc.acme.com";
int port = 16807;
IDictionary options = new SortedList();

options.Add(StorageConnector.OptionUserName, "MyUser");
options.Add(StorageConnector.OptionComputerName, "MyComputer");
options.Add(StorageConnector.OptionApplicationName, "MyApplication");

Storage stg = sc.OpenIMiSARCStorage(host, port, options);
```

Java

```
StorageConnector sc = IMIS_STORAGE_CONNECTOR;
String host = "iarc.acme.com";
int port = 16807;
Map options = new TreeMap();

options.put(StorageConnector.OPTION_USER_NAME, "Test User");
options.put(StorageConnector.OPTION_COMPUTER_NAME, "Test Computer");
options.put(StorageConnector.OPTION_APPLICATION_NAME, "Test Application");

Storage stg = sc.openIMiSARCStorage(host, port, options);
```

An example of sending a user message for the audit trail when opening an object on the archive server:

.NET

```
Storage stg = IMIS_ARCHIVE_V7;
string objectId = "e1aeed50688b8fd6df2b1aa93a8bd08620b7332561d84016b80428b69fe45e49";
string message = "Revision of scanned document %s";
System.Collections.IList arguments = new ArrayList();
arguments.Add("Invoice 1234");

stg.AuditLog.Message = message;
stg.AuditLog.Arguments = arguments;

Document doc = stg.OpenObject(objectId, DocumentAccess.Read);
```


Java

```
Storage stg = IMIS_ARCHIVE_V7;
String objectId = "e1aeed50688b8fd6df2b1aa93a8bd08620b7332561d84016b80428b69fe45e49";
String message = "Revision of scanned document %s";
java.util.List arguments = new ArrayList();
argumentList.Add("Invoice 1234");

stg.AuditLog.Message = message;
stg.AuditLog.Arguments = arguments;

Document doc = stg.OpenObject(objectId, Document.MODE_READONLY);
```

5.4.4 Examples for IMiS®/ARChive Server version 9

This chapter will present some of the most frequent examples of ways IMiS®/Storage Connector is used with IMiS®/ARChive Server version 9. These are operations for opening the archive, creating entities, reading and editing metadata and content in the form of files and providing data for the audit trail.

5.4.4.1 Opening the archive

To open the archive, the following are needed: a StorageConnector instance *(see chapter 5.4.1 Initializing IMiS®/Storage Connector)*, the network address of the server (host name or IP address) and a suitable port. To use the entity model on IMiS®/ARChive Server version 9 UserCredentials user authentication is required, which is why a collection of additional option parameters must be created. The collection will be sent to the method for opening the archive.

An example of opening the archive with UserCredentials authentication:

.NET

```
StorageConnector sc = IMIS_STORAGE_CONNECTOR;
string host = "iarc.acme.com";
int port = 16807;
IDictionary options = new SortedList();

options.Add(StorageConnector.OptionAuthType, AuthType.UserCredentials);
options.Add(StorageConnector.OptionUserName, "User1");
options.Add(StorageConnector.OptionUserPassword, "Password1");

IArchive arc = sc.OpenArchive(ArchiveType.IMiSARChive, host, port, options);
```


5.4.4.2 Public data about root classes

Retrieving public data about root classes on the archive through the IMiS®/Storage Connector interface takes place in the following steps:

- Obtaining a *StorageConnector* instance *(see chapter 5.4.1 Initializing IMiS®/Storage Connector)*.
- Opening the archive with a network address (host name or IP address), a port and user authentication *(see chapter 5.4.4.1 Opening the archive)*.
- Retrieving public data about the root classes of the archive using the method in the *IArchive* interface.

An example of retrieving public data without collection sorting:

.NET

```
IArchive arc = IMIS_ARCHIVE_V9;

ILargeReadOnlyList<IEntityStub> stubs = arc.GetRootClasses(null);
```

An example of retrieving public data with collection sorting by classification code:

.NET

```
IArchive arc = IMIS_ARCHIVE_V9;
IList<EntitySortKey> sortKeys = new List<EntitySortKey> {
EntitySortKey.GetSystemPropertySortKey(SystemProperty.ClassificationCode,
EntitySortKeyDirection.Ascending)
};

ILargeReadOnlyList<IEntityStub> stubs = arc.GetRootClasses(sortKeys);
```

5.4.4.3 Public data about an entity

Getting public data about an entity on the archive through the IMiS®/Storage Connector interface takes place in the following steps:

- Obtaining a *StorageConnector* instance *(see chapter 5.4.1 Initializing IMiS®/Storage Connector)*.
- Opening the archive with a network address (host name or IP address), a port and user authentication *(see chapter 5.4.4.1 Opening the archive)*.
- Getting public data about an entity identified with an identifier or classification code using the method in the *IArchive* interface.

An example of getting public data about a folder:

.NET

```
IArchive arc = IMIS_ARCHIVE_V9;
string classificationCode = FOLDER_CLASS_CODE;

IEntityStub stub = arc.GetEntityInfo(EntityIdKind.ClassificationCode, classificationCode);
```

5.4.4.4 Public data about subentities

Viewing subentities through the IMiS®/Storage Connector interface takes place in the following steps:

- Obtaining a *StorageConnector* instance *(see chapter 5.4.1 Initializing IMiS®/Storage Connector)*.
- Opening the archive with a network address (host name or IP address), a port and user authentication *(see chapter 5.4.4.1 Opening the archive)*.
- Getting public data about the selected entity on the archive through an identifier or a classification code *(see chapter 5.4.4.3 Public data about an entity)*.
- Viewing the public data of the child or subentities using the method in the *IEntityStub* interface.

An example of retrieving public data about subentities in a folder without collection sorting:

.NET

```
IEntityStub stub = FOLDER_STUB;

ILargeReadOnlyList<IEntityStub> stubs = stub.GetSubEntities(null);
```

An example of retrieving public data about subentities with collection sorting by classification code:

.NET

```
IEntityStub stub = FOLDER_STUB;
IList<EntitySortKey> sortKeys = new List<EntitySortKey> {
EntitySortKey.GetSystemPropertySortKey(SystemProperty.ClassificationCode,
EntitySortKeyDirection.Ascending)
};

ILargeReadOnlyList<IEntityStub> stubs = stub.GetSubEntities (sortKeys);
```


5.4.4.5 Creating an entity

Creating an entity on the archive through the IMiS®/Storage Connector interface takes place in the following steps:

- Obtaining a *StorageConnector* instance *(see chapter 5.4.1 Initializing IMiS®/Storage Connector)*.
- Opening the archive with a network address (host name or IP address), a port and user authentication *(see chapter 5.4.4.1 Opening the archive)*.
- Creating an entity using the method in the *IArchive* interface or a specialized *IEntity* interface for a template listed with a unique template identifier.
- Setting required system metadata values such as the name of the entity.
- Saving the entity on the archive.

An example of creating a class below the root archive using the *IArchive* interface:

.NET

```
IArchive arc = IMIS_ARCHIVE_V9;
string templateId = ROOT_CLASS_TEMPLATE;

IClass cls = arc.CreateClass(templateId);
cls.Title = "A root class";
cls.Save();
```

An example of creating a class below a class using the *IArchive* interface:

.NET

```
IArchive arc = IMIS_ARCHIVE_V9;
string templateId = CLASS_TEMPLATE_ID;
string parentClassificationCode = PARENT_CLASS_CODE;

IClass cls = arc.CreateClass(templateId, EntityIdKind.ClassificationCode, parentClassificationCode);
cls.Title = "Class in class";
cls.Save();
```

An example of creating a folder below a class using the *IClass* interface:

.NET

```
IClass cls = CLASS_ENTITY;
string templateId = FOLDER_TEMPLATE_ID;

IFolder folder = cls.CreateFolder(templateId);
folder.Title = "Folder in class";
folder.Save();
```


An example of creating a document in a folder using the *IFolder* interface:

.NET

```
IFolder folder = FOLDER_ENTITY;
string templateId = DOCUMENT_TEMPLATE;

IDocument document = folder.CreateDocument(templateId);
document.Title = "Document in folder";
document.Save();
```

5.4.4.6 Opening an entity

Opening an entity on the archive through the IMiS®/Storage Connector interface takes place in the following steps:

- Obtaining a *StorageConnector* instance *(see chapter 5.4.1 Initializing IMiS®/Storage Connector)*.
- Opening the archive with a network address (host name or IP address), a port and user authentication *(see chapter 5.4.4.1 Opening the archive)*.
- Opening an entity identified by an identifier or classification code using the method in *IArchive* for the selected access mode (read only or read and write) or opening an entity through the *IEntityStub* interface.

Example of opening a document in read only mode using the *IArchive* interface:

.NET

```
IArchive arc = IMIS_ARCHIVE_V9;
string classificationCode = DOCUMENT_CLASS_CODE;

IDocument document = arc.OpenDocument(EntityIdKind.ClassificationCode, classificationCode,
EntityAccess.Read);
```

Example of opening a document in read only mode using the *IEntityStub* interface:

.NET

```
IEntityStub stub = DOCUMENT_STUB;

IDocument document = stub.Open(EntityAccess.ReadWrite);
```


5.4.4.7 Moving an entity

Moving an entity on the archive with the IMiS®/Storage Connector interface takes place in the following steps:

- Obtaining a *StorageConnector* instance *(see chapter 5.4.1 Initializing IMiS®/Storage Connector)*.
- Opening the archive with a network address (host name or IP address), a port and user authentication *(see chapter 5.4.4.1 Opening the archive)*.
- Moving an entity to the selected parent entity (both entities (child and parent) are identified with their identifiers or classification codes) using the method in the *IArchive* interface or moving an entity using the the *IEntityStub* interface by giving the *IEntityStub* interface of the parent entity.

An example of moving a document using the *IArchive* interface:

.NET

```
IArchive arc = IMIS_ARCHIVE_V9;
string classificationCode = DOCUMENT_CLASS_CODE;
string parentClassificationCode = TARGET_FOLDER_CLASS_CODE;

arc.MoveEntity(EntityIdKind.ClassificationCode, classificationCode, EntityIdKind.ClassificationCode,
parentClassificationCode);
```

An example of moving a document using the IEntityStub interface:

.NET

```
IEntityStub stub = DOCUMENT_STUB;
IEntityStub parentStub = TARGET_FOLDER_STUB;

stub.Move(parentStub);
```

5.4.4.8 Deleting an entity

Deleting an entity on the archive through the IMiS®/Storage Connector interface takes place in the following steps:

- Obtaining a *StorageConnector* instance *(see chapter 5.4.1 Initializing IMiS®/Storage Connector)*.
- Opening the archive with a network address (host name or IP address), a port and user authentication *(see chapter 5.4.4.1 Opening the archive)*.
- Deleting an entity identified with an identifier or classification code from the archive using the method in the *IArchive* interface or using the *IEntityStub* interface.

An example of deleting a document using the *IArchive* interface:

.NET

```
IArchive arc = IMIS_ARCHIVE_V9;
string classificationCode = DOCUMENT_CLASS_CODE;

arc.DeleteEntity(EntityIdKind.ClassificationCode, classificationCode);
```

An example of deleting a document using the *IEntityStub* interface:

.NET

```
IEntityStub stub = DOCUMENT_STUB;

stub.Delete();
```

5.4.4.9 Viewing and editing data about an entity

Viewing and editing data about an entity on the archive through the IMiS®/Storage Connector interface takes place in the following steps:

- Obtaining a *StorageConnector* instance *(see chapter 5.4.1 Initializing IMiS®/Storage Connector)*.
- Opening the archive with a network address (host name or IP address), a port and user authentication *(see chapter 5.4.4.1 Opening the archive)*.
- Opening the entity on the archive with an identifier or a classification code *(see chapter 5.4.4.6 Opening an entity)*.
- Viewing and editing metadata values associated with the unique name of the entity.
- Saving any changes made to the content of the entity using the Save method in the entity.

An example of viewing and editing the metadata of an entity:

.NET

```
IEntityStub doc = DOCUMENT_ENTITY;
string propertyName = CUSTOM_STRING_PROPERTY_NAME;

string propertyValue = doc.Properties[propertyName].GetValue();
System.Console.WriteLine(propertyValue);

doc.Properties[propertyName].SetValue("New custom value");
doc.Save();

propertyValue = doc.Properties[propertyName].GetValue();
System.Console.WriteLine(propertyValue);
```


5.4.4.10 Saving records to a document

Saving a file to the content of a document on the archive using the IMiS®/Storage Connector interface takes place in the following steps:

- Obtaining a *StorageConnector* instance *(see chapter 5.4.1 Initializing IMiS®/Storage Connector)*.
- Opening the archive with a network address (host name or IP address), a port and user authentication *(see chapter 5.4.4.1 Opening the archive)*.
- Opening the entity on the archive with an identifier or a classification code *(see chapter 5.4.4.6 Opening an entity)*.
- Creating a file on the archive using the method in the *IArchive* interface and providing the suitable content type (MIME).
- Copying the file content to the archive using the data stream.
- Assigning the file to the content of a document.
- Saving any changes made to the content of the entity using the *Save* method in the document.

An example of saving a file in the content of a document:

.NET

```
IDocument doc = DOCUMENT_ENTITY;
string fileName = "c:\\Temp\\test.tif ";
string contentType = "image/tiff";

IContentPart contentPart = doc.CreateContentPart(contentType);
Stream contentPartStream = contentPart.OpenDataStream(EntityAccess.ReadWrite);
try {
  FileStream fileStream = new FileStream(fileName, FileMode.Open);
  try {
    int len;
    byte[] buffer = new byte[8192];
    while (0 < (len = fileStream.Read(buffer, 0, buffer.Length)))
      contentPartStream.Write(buffer, 0, len);
    contentPartStream.Flush();
  }
  finally {
    fileStream.Close();
  }
}
finally {
  contentPartStream.Close();
}
```


.NET (continued)

```
IList<IContentPart> contentParts = doc.Content.GetParts();
contentParts.Add(contentPart);
content.SetParts(contentParts);

doc.Save();
```

5.4.4.11 Providing data for the audit trail

The audit trail is a feature of IMiS®/ARChive Server that creates a record of operations with objects.

If this functionality is enabled on the server, clients are required to provide audit information. A selection of option parameters is then used to deliver this information when opening the archive.

The user name and computer name are of key importance for the audit trail. The name of the application is currently used only when recording operations on the archive server itself.

An example of opening the archive with audit trail information:

.NET

```
StorageConnector sc = IMIS_STORAGE_CONNECTOR;
string host = "iarc.acme.com";
int port = 16807;
IDictionary options = new SortedList();

options.Add(StorageConnector.OptionUserName, "MyUser");
options.Add(StorageConnector.OptionComputerName, "MyComputer");
options.Add(StorageConnector.OptionApplicationName, "MyApplication");

IArchive arc = sc.OpenArchive(ArchiveType.IMiSARChive, host, port, options);
```


An example of sending a user message for the audit trail when performing the opening operation on the archive server:

.NET

```
IArchive arc = IMIS_ARCHIVE_V9;
string classificationCode = DOCUMENT_CLASS_CODE;
string message = "Revision of scanned document %s";
System.Collections.IList arguments = new ArrayList();
arguments.Add("Invoice 1234");

stg.AuditLog.Message = message;
stg.AuditLog.Arguments = arguments;

IDocument doc = arc.OpenDocument(EntityIdKind.ClassificationCode, classificationCode);
```

5.4.4.12 Public data of reviews

Retrieving public data of a review on the archive through the IMiS®/Storage Connector interface takes place in the following steps:

- Obtaining a *StorageConnector* instance *(see chapter 5.4.1 Initializing IMiS®/Storage Connector)*.
- Opening the archive with a network address (host name and IP address), a port and user authentication *(see chapter 5.4.4.1 Opening the archive)*.
- Retrieving public data about all or only selected reviews, identified with an identifier or classification code using the method in the *IArchive* interface.

An example of retrieving public data of all reviews without collection sorting:

.NET

```
IArchive arc = IMIS_ARCHIVE_V9;

ILargeReadOnlyList<IReviewStub> reviewStubs = arc.GetReviews(null);
```

An example of retrieving public data about all reviews with collection sorting by classification code:

.NET

```
IArchive arc = IMIS_ARCHIVE_V9;
IList<EntitySortKey> sortKeys = new List<EntitySortKey> {
EntitySortKey.GetSystemPropertySortKey(SystemProperty.ClassificationCode, EntitySortKeyDirection.Ascending)
};
ILargeReadOnlyList< IReviewStub > reviewStubs = arc.GetReviews(sortKeys);
```


An example of retrieving public data about a selected review:

.NET

```
IArchive arc = IMIS_ARCHIVE_V9;
string classificationCode = REVIEW_CLASS_CODE;

IReviewStub reviewStub = arc.GetReviewInfo(EntityIdKind.ClassificationCode, classificationCode);
```

5.4.4.13 Creating a review

Creating a review on the archive through the IMiS®/Storage Connector interface takes place in the following steps:

- Obtaining a *StorageConnector* instance *(see chapter 5.4.1 Initializing IMiS®/Storage Connector)*.
- Opening the archive with a network address (host name or IP address), a port and user authentication *(see chapter 5.4.4.1 Opening the archive)*.
- Creating a review through one of the methods on the *IArchive* interface.
- Setting required system metadata values such as the address and members of the commission in the review;
- Saving the review on the archive.

An example of creating a review with selected retention policies through the *IArchive* interface:

.NET

```
IArchive arc = IMIS_ARCHIVE_V9;
string templateId = ROOT_CLASS_TEMPLATE;
IList<string> members = MEMBERS;
IEntityStub scopeEntity = SCOPE_ENTITY;
ICollection<IRetentionPolicy> retentionPolicies = RETENTION_POLICIES;

IReview review = arc.CreateReview(scopeEntity, retentionPolicies);
review.Title = "A review";
review.Members = members;
review.Save();
```


An example of creating a review with a selected query through the *IArchive* interface:

.NET

```
IArchive arc = IMIS_ARCHIVE_V9;
string templateId = ROOT_CLASS_TEMPLATE;
IList<string> members = MEMBERS;
IEntityStub scopeEntity = SCOPE_ENTITY;
string queryExporession = QUERY_EXPRESSION;

IReview review = arc.CreateReview(scopeEntity, queryExporession);
review.Title = "A review";
review.Members = members;
review.Save();
```

5.4.4.14 Opening a review

Opening a review on the archive through the IMiS®/Storage Connector interface takes place in the following steps:

- Obtaining a *StorageConnector* instance *(see chapter 5.4.1 Initializing IMiS®/Storage Connector)*.
- Opening the archive with a network address (host name or IP address), a port and user authentication *(see chapter 5.4.4.1 Opening the archive)*.
- Opening a review identified by an identifier or classification code using the method in *IArchive* for the selected access mode (read only or read and write) or opening a review through the *IReviewStub* interface.

An example of opening a review for reading through the *IArchive* interface:

.NET

```
IArchive arc = IMIS_ARCHIVE_V9;
string classificationCode = REVIEW_CLASS_CODE;

IReview review = arc.OpenReview(EntityIdKind.ClassificationCode, classificationCode, EntityAccess.Read);
```

An example of opening a review for reading and writing through the *IReviewStub* interface:

.NET

```
IReviewStub reviewStub = DOCUMENT_STUB;

IReview review = reviewStub.Open(EntityAccess.ReadWrite);
```


5.4.4.15 Searching the archive

IMiS/ARChive Server version 9 enables searching by entity metadata and/or the full text of archived records. Searching can be performed on the entire archive or just in a selected entity in the classification scheme. Searching the archive using the IMiS®/Storage Connector interface takes place in the following steps:

- Obtaining a *StorageConnector* instance *(see chapter 5.4.1 Initializing IMiS®/Storage Connector)*.
- Opening the archive with a network address (host name or IP address), a port and user authentication *(see chapter 5.4.4.1 Opening the archive)*.
- Viewing public data about an entity *(see chapter 5.4.4.3 Public data about an entity)* or using an identifier or classification code to open the entity *(see chapter 5.4.4.6 Opening an entity)* under which the user would like to perform the search.
- Searching the archive using the method in the *IArchive* interface or searching below the selected entity using the method in the *IEntityStub* or *IEntity* interface.

An example of searching the entire archive for classes and folders whose names start with the letter A:

.NET

```
IArchive arc = IMIS_ARCHIVE_V9;
EntityFilter filter = EntityFilter.Classes | EntityFilter.Folders;
ICollection<EntitySortKey> sortKeys = null;
SearchOptions options = SearchOptions.Recursive;
string expression = String.Format("[{0}]=\"{1} \"",
arc.GetSystemPropertyName(SystemProperty.Title), "A*");

ILargeReadOnlyList<IEntityStub> stubs = arc.Search(filter, sortKeys, options, expression);
```

An example of searching for documents containing the word “invoice” below a specified folder:

.NET

```
IEntityStub stub = FOLDER_STUB;
EntityFilter filter = EntityFilter.Documents;
ICollection<EntitySortKey> sortKeys = new List<EntitySortKey> {
EntitySortKey.GetSystemPropertySortKey(SystemProperty.Title, EntitySortKeyDirection.Ascending)
};
SearchOptions options = SearchOptions.Recursive;
string expression = String.Format("{{{0}}}", "Invoice");

ILargeReadOnlyList<IEntityStub> stubs = stub.Search(filter, sortKeys, options, expression);
```


An example of searching the metadata and full text of documents below a specified folder for documents whose title starts with the letter A and that contain the word “invoice”:

.NET

```
IArchive arc = IMIS_ARCHIVE_V9;
IEntityStub stub = FOLDER_STUB;
EntityFilter filter = EntityFilter.Documents;
ICollection<EntitySortKey> sortKeys = new List<EntitySortKey> {
EntitySortKey.GetSystemPropertySortKey(SystemProperty.Title, EntitySortKeyDirection.Ascending)
};
SearchOptions options = SearchOptions.Recursive;
string expression = String.Format("[{0}]=\"{1} \" AND {{{2}}}",
arc.GetSystemPropertyName(SystemProperty.Title), "A*", "Invoice");

ILargeReadOnlyList<IEntityStub> stubs = stub.Search(filter, sortKeys, options, expression);
```

5.4.5 Logging IMiS®/Storage Connector

Logging or creating records of the operations of IMiS®/Storage Connector is an important function for administrators and application developers. It serves to monitor operations and pinpoint and solve problems that may arise during use.

The interface supports two logging methods. The first is so-called internal logging to a rotating log file at a specified location with records in a predefined form. The second method is based on the specific needs of the user. The user can specify the handlers of records (a log file, console window, etc.) and customize the form of records to suit their needs (for example, records can be in plain text format or XML format).

5.4.5.1 Internal logging

IMiS®/Storage Connector enables internal logging, which means that the program sees to the creation of records on its operations. These records are saved to a rotating log file at a predefined location in the file system.

Internal logging is supported through a *StorageConnector* instance (*see chapter 5.4.1 Initializing IMiS®/Storage Connector*) and takes place in the following way:

.NET

```
StorageConnector sc = IMIS_STORAGE_CONNECTOR;

sc.LogInternal = true;
```


Java
StorageConnector sc = IMIS_STORAGE_CONNECTOR; sc.logInternal(true);

If the IMiS®/Storage Connector .NET interface is being used, the log file is rotating and is located in the system temporary directory. The name of the log files is equal to *IMiS.StorageConnector.NET.X.log*, where X represents a generated number separating the different rotating log files.

A rotating log file is also used with IMiS®/Storage Connector Java. The name of the log files is equal to *IMiS.StorageConnector.Java.X.log*, where X represents a generated number separating the different log files. Location:

- On Windows systems it is located in the system temporary directory (the path is returned by the *System.getProperty("java.io.tmpdir")* method).
- Otherwise it is located in the user's home directory (the path is returned by the *System.getProperty("user.home")* method.

The number of rotating log files is limited to 10, and each file has a size limit of roughly 1 MB.

5.4.5.2 Custom logging

Users can customize logging in IMiS®/Storage Connector to suit their specific needs by using their own handlers for creating records on product operations. The pre-implemented handlers in the .NET Framework or Java can be used, or the user can implement their own handlers.

If the IMiS®/Storage Connector .NET version is being used, the handlers must be based on the *System.Diagnostics.TraceListener* abstract class. Additional information about this class is available on Microsoft's website:
http://msdn.microsoft.com/en-us/library/system.diagnostics.tracelistener(v=vs.80).aspx
for .NET 2.0
http://msdn.microsoft.com/en-us/library/system.diagnostics.tracelistener(v=vs.90).aspx
for .NET 3.5
http://msdn.microsoft.com/en-us/library/system.diagnostics.tracelistener(v=vs.100).aspx
for .NET 4.0

Some handlers are already implemented in the .NET Framework, including:

- *System.Diagnostics.TextWriterTraceListener*, which creates a record to a file or to the data stream.
- *System.Diagnostics.ConsoleTraceListener*, which writes to the console window.
- *System.Diagnostics.EventLogTraceListener*, which writes to the system event log.

A handler in IMiS®/Storage Connector .NET can also be based on the *IMiS.Diagnostics.LogHandler* abstract class. This class derives from the *System.Diagnostics.TraceListener* class and provides additional options for customizing reports. It is located in the imisbase.net.dll library, which is part of IMiS®/Storage Connector .NET.

An example of implementing a handler derived from the *IMiS.Diagnostics.LogHandler* class would be *IMiS.Diagnostics.FileLogHandler*. This will write to a file and will also be used in internal logging *(see chapter 5.4.5.1 Internal logging)*.

If the IMiS®/Storage Connector Java version is being used, the handler must derive from the *java.util.logging.Handler* abstract class. Additional information about this class is available on Oracle's website:
http://docs.oracle.com/javase/1.4.2/docs/api/java/util/logging/Handler.html

Examples of handlers implemented in Java include:

- *java.util.logging.FileHandler*, which writes to a file.
- *java.util.logging.ConsoleHandler*, which writes to the console window.
- *java.util.logging.SocketHandler*, which writes over the network.

Custom logging is enabled through the *StorageConnector* instance *(see chapter 5.4.1 Initializing IMiS®/Storage Connector)*; the administrator creates a handler and adds it to the list of handlers.

An example of creating a handler that writes to a file:

.NET

```
        StorageConnector sc = IMIS_STORAGE_CONNECTOR;
string fileName = @"c:\isc.log";
        IMiS.Diagnostics.FileLogHandler fileLog = new FileLogHandler(fileName, "My File Log", 1000000, 10,
        FileLogOptions.Append);

sc.LogHandlers.Add(fileLog);
```

Java

```
        StorageConnector sc = IMIS_STORAGE_CONNECTOR;
String fileName = "/isc.log";
java.util.logging.FileHandler fileLog = new FileHandler(fileName, 1000000, 10, true);

sc.logAddHandler(fileLog);
```

5.4.5.3 Logging levels

IMiS®/Storage Connector enables different levels of logging. The Administrator can set limits on which types of records of operational events are created in the log file or in some other handler.

Example: An administrator can choose to record only errors and information about the system (the default setting) or they can choose to keep records on even the most minute aspects of system operations.

Generally speaking, less logging means, among other things, faster logging. However, in the event that problems arise in system use it might be beneficial to change the level of logging. It is a good idea to set logging to the most detailed level and to repeat the events that led to the issue.

If an administrator cannot successfully resolve the issues by analyzing the log, we advise him or her to send it, along with a description of the problem, to *support@imis.eu*.

The following constants are used to set the logging level:

.NET	
Property	**Description**
SourceLevels.Off	Logging disabled.
SourceLevels.Critical	Logging level that only records critical errors.
SourceLevels.Error	Logging level that records common and critical errors.
SourceLevels.Warning	Logging level that records warnings and common and critical errors.
SourceLevels.Information	Logging level that records informational records, warnings and common and critical errors.
SourceLevels.Verbose	Logging level that records details on operations, informational records, warnings and common and critical errors.
SourceLevels.All	Logging level that records all events.

Java	
Method	**Description**
Level.OFF	Logging disabled.
Level.SEVERE	Logging level that only records errors.
Level.WARNING	Logging level that records warnings and errors.
Level.INFO	Logging level that records informational records, warnings and errors.
Level.FINE	Logging level that records details on operations, informational records, warnings and common and critical errors.
Level.FINER	Logging level that records details on operations, informational records, warnings and common and critical errors.
Level.FINEST	Logging level that records details on operations, informational records, warnings and common and critical errors.
Level.ALL	Logging level that records all events.

The logging level is set using the *StorageConnector* instance *(see chapter 5.4.1 Initializing IMiS®/Storage Connector)*. One of the constants listed above is used, as the example below shows.

Example of setting logging to the most detailed level:

.NET

```
        StorageConnector sc = IMIS_STORAGE_CONNECTOR;

sc.LogLevel = System.Diagnostics.SourceLevels.Verbose;
```

Java

```
        StorageConnector sc = IMIS_STORAGE_CONNECTOR;

sc.logSetLevel(java.util.logging.Level.FINEST);
```


6 TROUBLESHOOTING

6.1 Problems using IMiS®/Storage Connector .NET version

Below, issues frequently encountered when using IMiS®/Storage Connector are described and instructions for resolving them are given.

6.1.1 Issues with references in a development project

When developing applications with IMiS®/Storage Connector .NET usually a reference to the *storageconnector.net.dll* library is not sufficient.

Example: If an administrator in a development project would like to catch errors specific to the interface, when attempting to create a binary code (build)...

.NET

```
try {
  ...
}
catch (StorageConnectorException ex) {
  ...
}
```

...the following error occurs:

1) The type 'IMiS.GlobalizedException' is defined in an assembly that is not referenced. You must add a reference to assembly 'imisbase.net, Version=2.0.0.0, Culture=neutral, PublicKeyToken=51833a6f82ea576f'
2) The type caught or thrown must be derived from System.Exception

Cause of the problem: In a development project, the development environment requires a reference to the library containing the unknown class in the error description.
In this example, the unknown class is defined in the *misbase.net.dll* library, which is not listed in the references of the development project.
This class is a basic class for all types of errors in IMiS® software on the .NET platform.

System.Exception
IMiS.BaseException
IMiS.GlobalizedException
IMiS.StorageConnector.StorageConnectorException

Resolving the problem: a reference to the *imisbase.net.dll* library must be added to the references in the development project, as this library is also a part of the interface.

6.2 Problems using IMiS®/Storage Connector Java version

6.2.1 Issues with references in a development project

When developing applications with IMiS®/Storage Connector Java usually a reference to the *storageconnector.jar* library in the *classpath* of the development project is not sufficient.
If the application developer is using a method that can return an error specific to this interface, as shown below...

Java

```
static Storage openIMiSARCStorage(String host, int port)
  throws StorageConnectorException
{
  // opening the archive with an instance of IMiS/Storage Connector Java
  return sc.openIMiSARCStorage(host, port);
}
```

...the following error occurs:

No exception of type StorageConnectorException can be thrown; an exception type must be a subclass of Throwable

Cause of the problem: In a development project, the development environment requires a reference to the library containing the unknown class in the error description. In this example, this class is the basic class for all types of errors in IMiS® software on the Java platform.
This class is specified in the *imisbase.jar* library, which is not listed in the *classpath* of the development project.

java.lang.Throwable
java.lang.Exception
com.imis.GlobalizedException
com.imis.storageconnector.StorageConnectorException

Resolving the problem: A reference to the *imisbase.jar* library must be added to the classpath, as this library is also a part of the IMiS®/Storage Connector Java interface.

6.2.2 Issues with unhandled errors

If an application developer calls up an IMiS®/Storage Connector Java method that could return an error as shown below...

Java

```
static Storage openIMiSARCStorage(String host, int port)
{
  // opening the archive with an instance of IMiS/Storage Connector Java
  return sc.openIMiSARCStorage(host, port);
}
```

...the following error occurs:

Unhandled exception type StorageConnectorException

Cause of the problem: Java identifies two types of errors: checked and unchecked.
Checked errors are those errors that are not derived from the *java.lang.Error* or even the *java.lang.RuntimeException* classes and which must therefore be processed within a method or listed together with a method declaration.
In the above example, the error that could be caused by the called up method is a checked error, which is why Java tells the user the error is unhandled.

Resolving the problem: When working with a checked error, it must be listed either with a method declaration or with a *throws* sentence...

Java

```
static Storage OpenIMiSARCStorage(String host, int port)
  throws StorageConnectorException
{
  // opening the archive with an instance of IMiS/Storage Connector Java
  return sc.openIMiSARCStorage(host, port);
}
```


...or handled within the method:

Java

```
static Storage OpenIMiSARCStorage(String host, int port)
{
  Storage stg = null;
  try {
    // opening the archive with an instance of IMiS/Storage Connector Java
    stg = sc.openIMiSARCStorage(host, port);
  }
  catch (StorageConnectorException e) {
    // ..
  }
  return stg;
}
```

6.2.3 Problems opening a session between the server and a client

When IMiS®/Storage Connector Java is running, an error may occur when setting up a session between IMiS®/ARChive Server and a client using the interface.

Operations on the archive where this issue could occur are creating, opening or deleting objects.

The issue is logged in the log as follows:

Java

```
com.imis.storageconnector.StorageConnectorException: Error occurred while opening session on IMiS/ARC
Server <iarc.acme.com:16807>
...
com.imis.imisarc.client.IAClientException: Authentication between server and client failed.
...
```

Cause of the problem: The issue could be caused by improper *Java Cryptography Extension (JCE)* policy files, which usually do not have the required 192/256-byte AES encryption if the *Java Runtime Environment (JRE)* was installed normally due to legal constraints in the US regarding the export of cryptographic products.

Resolving the problem: *JCE policy* files need to be upgraded to *Java JVM*.

With Sun Microsystems JRE, *JCE policy* files are available on Oracle's website:

http://www.oracle.com/technetwork/java/javasebusiness/downloads/java-archive-downloads-java-plat-419418.html#7503-jce-1.4.2-oth-JPR

If IBM JRE is used, the appropriate IBM *JCE policy* files can be found on IBM's website:

https://www14.software.ibm.com/webapp/iwm/web/preLogin.do?source=jcesdk

The *JCE policy* files replace existing files in the JRE Security directory...

<java-home>/lib/security [in Java 2 Runtime Environment]
<java-home>/jre/lib/security [in Java 2 SDK]

...where *<java-home>* represents the directory where JRE or JDK is installed.

6.2.4 Problems with log editing rights

When activating IMiS®/Storage Connector Java, a problem may occur due to restrictions in the rights settings in the Java environment.

The problem is logged as follows in the system log:

Java
java.security.AccessControlException: access denied (java.util.logging. LoggingPermission control) at java.security.AccessControlContext.checkPermission(AccessControlContext.java:269) at java.security.AccessController.checkPermission(AccessController.java:401) at java.lang.SecurityManager.checkPermission(SecurityManager.java:524) at java.util.logging.LogManager.checkAccess(LogManager.java:834) at java.util.logging.Handler.checkAccess(Handler.java:276) at java.util.logging.FileHandler.<init>(FileHandler.java:329) at com.imis.storageconnector.StorageConnectorLogger.setInternalLogging(StorageConnectorLogger.java:174) at com.imis.storageconnector.StorageConnector.logInternal(StorageConnector.java:608) ...

Cause of the problem: The problem pertains to excluded rights for logging in the Java environment. Specifically, the *java.util.logging.LoggingPermission* right has not been included in the *Java Authentication and Authorization Service (JAAS) policy* file.

Resolving the problem: The system or user policy file must be fixed to include the right or permission to write in the log (*java.util.logging.LoggingPermission*).

The example below shows how to resolve internal logging problems in a project that uses IMiS®/Storage Connector Java in a Windows environment where the system policy file is replaced with a user file.

To successfully write in the log, which is created in the system temporary directory on Windows systems (*see "Internal logging"*), besides the right to write in the log the right to read the *java.io.tmpdir* system values and the right to read, write and delete in the system temporary directory must be specified.

Java

```
grant codeBase "file:/C:/IMiS/iscjavatest/-" {
  permission java.util.PropertyPermission "java.io.tmpdir", "read";
        permission java.io.FilePermission "${java.io.tmpdir}${/}-", "read,write,delete";
  permission java.util.logging.LoggingPermission "control";
};
```

Additional information on permissions and policy files in Java is available on Oracle's website at the following links: *http://docs.oracle.com/javase/1.4.2/docs/*

6.3 List of errors that may occur when using IMiS®/Storage Connector

Below, frequent errors that could occur when using the IMiS®/Storage Connector interface are listed. Each description is followed by the cause of the error and instructions on how to deal with the error.

6.3.1 Errors in IMiS®/ARChive Server 7

Demo license expired.

This notification means that the demo version of IMiS®/Storage Connector .NET or Java has expired. To renew the demo version, contact *info@imis.eu*.

Feature/method is currently not supported.

This notification means that the feature or method that returned the error is currently not supported or implemented.

Error occurred while opening a session.

This notification informs the user that an error occurred when attempting to establish a connection with IMiS®/ARChive Server.

The cause of the error could be an incorrectly listed host name or port, an error determining the IP address from the host name, a network connection error, an unresponsive server, an invalid server response, an error sent by the server when establishing the connection or failed authentication between the server and the client. This error is critical, and there is a chance that the product manufacturer will need to intervene. We advise users to promptly send a log of the error to the manufacturer at *support@imis.eu*.

Error occurred while creating an object on IMiS/ARC Server <*host*> (profile=<*profile_name*> mime=<*mime_type*>).

This notification tells the user that an error occurred when attempting to create an object on IMiS®/ARChive Server using *Storage.CreateObject()* (.NET) or *Storage.createObject()* (Java). The cause of the error could be an incorrect profile or MIME type, an interrupted connection, an error sent by the server when creating the object or an error associated with the server's compression library. This error is critical, and there is a chance that the product manufacturer will need to intervene.

We advise users to promptly send a log of the error to the manufacturer at *support@imis.eu*.

Error occurred while opening object <*object_identifier*> on IMiS®/ARC Server <*host*> in mode <*access_mode*>.

This notification tells the user that an error occurred when attempting to open an object on IMiS®/ARChive Server using *Storage.OpenObject()* (.NET) or *Storage.openObject()* (Java). The cause of the error could be an incorrect object identifier, an interrupted connection, an error sent by the server when opening the object or an error associated with the server's compression library. This error is critical, and there is a chance that the product manufacturer will need to intervene. We advise users to promptly send a log of the error to the manufacturer at *support@imis.eu*.

Error occurred while committing object <*object_identifier*> changes on IMiS/ARC Server <*host*>.

This notification means that an error occurred when saving an object on IMiS®/ARChive Server using the *Document.Save()* (.NET) or *Document.save()* (Java) method. The error could be caused by an interrupted connection, a connection error or an error on the server, or the object could already be closed. It would be a good idea to promptly send a log of the error to the manufacturer at *support@imis.eu* for further analysis.

Error occurred while moving object <*object_identifier*> on IMiS/ARC Server <*host*> to new profile <*profile_name*>.

This notification tells the user that an error occurred when attempting to move an object on IMiS®/ARChive Server using *Storage.MoveObject()* (.NET) or *Storage.moveObject()* (Java). This error occurs because this feature is not supported.

Error occurred while removing object <*object_identifier*> on IMiS®/ARChive Server <*host*>.

This notification tells the user that an error occurred when attempting to delete an object on IMiS®/ARChive Server using *Storage.DeleteObject()* (.NET) or *Storage.deleteObject()* (Java). The cause of the error could be an incorrect object identifier, an interrupted connection or an error sent by the server when deleting the object. This error could represent a problem that requires the product manufacturer's intervention. We advise users to promptly send a log of the error to the manufacturer at *support@imis.eu*.

Error occurred while closing object <*object_identifier*> on IMiS/ARC Server <*host*>.

This notification means that an error occurred when closing an object on IMiS®/ARChive Server using the *Document.Close()* (.NET) or *Document.close()* (Java) method. The error could be caused by an interrupted connection or an error sent by the server when closing the object, or the object could already be closed. In theory this error does not represent a major problem; it would however be a good idea to send a log of the error to the manufacturer at *support@imis.eu* for further analysis.

Object identifier is not valid until new object is saved.

This notification means that an object identifier was requested through *Document.Id* (.NET) or *Document.getId()* (Java), but the object was just created on IMiS®/ARChive Server and has yet to be saved. An object does not have a valid identifier until it is saved. This error does not represent a serious problem. It is just a warning about correct program use.

Operation on closed object is not allowed.

This notification means that the object that returned this error when the method was called up was already closed using the *Document.Close()* (.NET) or *Document.close()* (Java) method. This error does not represent a serious problem. It is just a warning about correct program use.

Operation on closed storage is not allowed.

This notification means that the archive that returned this error when the method was called up was already closed using the *Storage.Close()* (.NET) or *Storage.close()* (Java) method. This error does not represent a serious problem. It is just a warning about correct program use.

File has no extension which is required if MIME type is not provided!

This notification tells the user that an error occurred when attempting to save an object on IMiS®/ARChive Server using *Storage.StoreObject()* (.NET) or *Storage.storeObject()* (Java). The error occurred because the provided file name does not have an extension and the MIME type was not provided. This error does not represent a serious problem. It is just a warning about correct program use. For help resolving this problem, the administrator can send a log of the error to the manufacturer at *support@imis.eu*.

Error getting object data stream.

This notification means that an error occurred when attempting to get the data stream of an object located on IMiS®/ARChive Server using *Document.DataStream* (.NET). The error could be caused by an interrupted connection or a problem with the server's compression library, or the object could already be closed. This error is critical, and there is a chance that the product manufacturer will need to intervene. We advise users to promptly send a log of the error to the manufacturer at *support@imis.eu*.

Error getting object input data stream.

This notification means that an error occurred when attempting to get a data stream for reading an object located on IMiS®/ARChive Server using *Document.DataStream* (Java). The error could be caused by an interrupted connection, improper simultaneous use of the data stream for writing or a problem with the server's compression library, or the object could already be closed. This error is critical, and there is a chance that the product manufacturer will need to intervene. We advise users to promptly send a log of the error to the manufacturer at *support@imis.eu*.

Error getting object output data stream.

This notification means that an error occurred when attempting to get a data stream for writing an object located on IMiS®/ARChive Server using *Document.getInputDataStream()* (Java). The error could be caused by an interrupted connection, improper simultaneous use of the data stream for reading or a problem with the server's compression library, or the object could already be closed. This error is critical, and there is a chance that the product manufacturer will need to intervene. We advise users to promptly send a log of the error to the manufacturer at *support@imis.eu*.

Error copying stream contents.

This notification means that an error occurred when copying an object's contents through the object's data stream. The cause could be an issue in the input data stream or in the data stream for writing the object on IMiS®/ARChive Server. The error represents an issue that should be addressed by the manufacturer. Send a log of the error and a description to *support@imis.eu*.

6.3.2 Errors for IMiS®/ARChive Server 9

Demo license expired.

This notification means that the test or demo version of IMiS®/Storage Connector .NET or Java has expired. To renew the demo version, contact *info@imis.eu*.

Error occurred while opening a session.

This notification informs the user that an error occurred when attempting to establish a connection with IMiS®/ARChive Server. The cause of the error could be an incorrectly listed host name or port, an error determining the IP address from the host name, a network connection error, an unresponsive server,
an invalid server response, or an error sent by the server when establishing a connection between the server and the client. This error is critical, and there is a chance that the product manufacturer will need to intervene. We advise users to promptly send a log of the error to the manufacturer at *support@imis.eu*.

User name or password for UserCredentials authentication is invalid.

This notification informs the user that an error occurred when attempting to establish a connection with IMiS®/ARChive Server. The error is caused by unsuccessful authentication between the server and the client. This error does not represent a serious problem. It is just a warning that incorrect information was sent during authentication (an incorrect user name or password).

Error occurred while getting available archives on server <*host*>.

This notification informs the user that an error occurred when attempting to get archives on IMiS®/ARChive Server. This error is sent by the server.
This error is critical, and there is a chance that the product manufacturer will need to intervene. We advise users to promptly send a log of the error to the manufacturer at *support@imis.eu*.

Error occurred while getting directory members on archive <*host*>.

This notification informs the user that an error occurred when attempting to get groups or users from the directory on IMiS®/ARChive Server. This error is sent by the server.
This error is critical, and there is a chance that the product manufacturer will need to intervene. We advise users to promptly send a log of the error to the manufacturer at *support@imis.eu*.

Error occurred while getting root classes on archive <*host*>.

This notification informs the user that an error occurred when attempting to get collections of root classes on IMiS®/ARChive Server. This error is sent by the server.
This error is critical, and there is a chance that the product manufacturer will need to intervene. We advise users to promptly send a log of the error to the manufacturer at *support@imis.eu*.

Error occurred while getting child entity stubs.

This notification informs the user that an error occurred when attempting to get collections of sub-entities on IMiS®/ARChive Server. This error is sent by the server.
This error is critical, and there is a chance that the product manufacturer will need to intervene. We advise users to promptly send a log of the error to the manufacturer at *support@imis.eu*.

Error occurred while getting templates on archive <*host*>.

This notification informs the user that an error occurred when attempting to get templates for entities on IMiS®/ARChive Server. This error is sent by the server. This error is critical, and there is a chance that the product manufacturer will need to intervene. We advise users to promptly send a log of the error to the manufacturer at *support@imis.eu*.

Error occurred while creating entity on archive <*host*>.

This notification informs the user that an error occurred when attempting to create an entity on IMiS®/ARChive Server. The cause of the error could be that the user does not have read rights for the parent entity, the parent entity has *Closed* status, or some other error sent by the server when creating the entity. This error is critical, and there is a chance that the product manufacturer will need to intervene. We advise users to promptly send a log of the error to the manufacturer at *support@imis.eu*.

Error occurred while opening entity <*entity_identifier*> on archive <*host*>.

This notification informs the user that an error occurred when attempting to open an entity on IMiS®/ARChive Server. The cause of the error could be that the user does not have read rights, the entity is already open in write mode, or some other error sent by the server when opening the entity. This error is critical, and there is a chance that the product manufacturer will need to intervene. We advise users to promptly send a log of the error to the manufacturer at *support@imis.eu*.

Operation on unsaved new entity is not allowed.

This notification means that an unpermitted operation has been performed on a new, unsaved entity. This error occurs when the user attempts to access data that is generated only once an entity has been successfully saved on IMiS®/ARChive Server.

For example, this error occurs when reading *Id*, *ClassificationCode*, or *PublicClassificationCode* or when using the *GetReport* method in the *IEntity* interface. This error does not represent a serious problem. It is just a warning about correct use of the program.

Operation on saved entity is not allowed.

This notification means that an unpermitted operation has been performed on a saved entity. This error occurs when a user attempts to edit information than can only be set when creating a new entity. For example, this error occurs when changing values for *Id*, *ClassificationCode*, *Status* or *SecurityClass* in the *IEntity* interface. This error does not represent a serious problem. It is just a warning about correct use of the program.

Operation on entity opened in read-only mode is not allowed.

This notification means that an unpermitted operation has been performed on an entity opened in read-only mode. This error occurs when a user attempts to edit information about an entity that is open in read-only mode, such as when a user attempts to set new values for *Title*, *Description* or *Owner* in the *IEntity* interface. This error does not represent a serious problem. It is just a warning about correct use of the program.

Operation on closed entity is not allowed.

This notification means that an unpermitted operation has been performed on a closed entity. This error occurs when operations are performed on an entity after the *Close* method has been called up in the *IEntity* interface. This error does not represent a serious problem. It is just a warning about correct use of the program.

Error occurred while committing entity changes on archive <*host*>.

This notification informs the user that an error occurred when attempting to save changes to an entity on IMiS®/ARChive Server. This error is sent by the server when saving an entity. This error is critical, and there is a chance that the product manufacturer will need to intervene. We advise users to promptly send a log of the error to the manufacturer at *support@imis.eu*.

Error occurred while moving entity <*entity_identifier*> on archive <*host*>.

This notification informs the user that an error occurred when attempting to move an entity on IMiS®/ARChive Server. The cause of the error could be that the user does not have write rights, that one of the parent entities or the target parent entity has *Closed* status, or some other error sent by the server when creating the entity. This error is critical, and there is a chance that the product manufacturer will need to intervene. We advise users to promptly send a log of the error to the manufacturer at *support@imis.eu*.

Error occurred while removing entity <*entity_identifier*> on archive <*host*>.

This notification informs the user that an error occurred when attempting to delete an entity on IMiS®/ARChive Server. The cause of the error could be that the user does not have delete rights, that one of the parent entities has Closed status, that the entity is of Vital or Permanent significance or some other error sent by the server when deleting the entity. This error is critical, and there is a chance that the product manufacturer will need to intervene. We advise users to promptly send a log of the error to the manufacturer at *support@imis.eu*.

Error occurred while setting entity property value.

This notification informs the user that an error occurred when attempting to set metadata values for an entity on IMiS®/ARChive Server. This error is sent by the server when setting metadata values. This error is critical, and there is a chance that the product manufacturer will need to intervene. We advise users to promptly send a log of the error to the manufacturer at *support@imis.eu*.

Error occurred while creating binary/file/string value.

This notification informs the user that an error occurred when attempting to create a binary/file/string value on IMiS®/ARChive Server. This error is sent by the server when creating a binary/file/string value. This error is critical, and there is a chance that the product manufacturer will need to intervene. We advise users to promptly send a log of the error to the manufacturer at *support@imis.eu*.

Error occurred while opening binary/file/string value data stream.

This notification informs the user that an error occurred when attempting to open a binary/file/string value data stream for the property of an entity on IMiS®/ARChive Server. This error is sent by the server when opening an entity. This error is critical, and there is a chance that the product manufacturer will need to intervene. We advise users to promptly send a log of the error to the manufacturer at *support@imis.eu*.

Property <*property_name*> is not present in entity's template <*template_identifier*>.

This notification informs the user that an error occurred when attempting to read or write the metadata of an entity on IMiS®/ARChive Server. The error is caused by an incorrect archive server configuration. This issue needs to be addressed by the manufacturer. We advise users to promptly send a log of the error to the manufacturer at *support@imis.eu*.

System property <*property_name*> is not present in entity's template <*template_identifier*>.

This notification informs the user that an error occurred when attempting to read or write the system metadata of an entity on IMiS®/ARChive Server. The error is caused by an incorrect archive server configuration. This issue needs to be addressed by the manufacturer.
We advise users to promptly send a log of the error to the manufacturer at *support@imis.eu*.

Generic argument type <*generic_type*> is not compatible with property type <*property_type*>.

This notification informs the user that an error occurred when attempting to read or write the metadata of an entity. This error is caused when a generic type which does not match the type of values in the metadata is used when calling up the *GetValue*/ *GetValues* or *SetValue*/ *SetValues* method in the *IReadOnlyProperty*/ *IProperty* interface. This error does not represent a serious problem. It is just a warning about correct program use.

Error occurred while performing search on archive/entity <*host*/ *entity_identifier*>.

This notification informs the user that an error occurred when attempting to search the entire archive or below the selected entity on IMiS®/ARChive Server. This error is sent by the server when searching the archive. This error is critical, and there is a chance that the product manufacturer will need to intervene. We advise users to promptly send a log of the error to the manufacturer at *support@imis.eu*.

