FILENET
IDL – Inbound Document Linker
Version 3.2
SIEBEL eBusiness
© 2004 FileNet Corporation. All Rights Reserved.

Operations Manual

Release 3.2

May 2004

FileNet Corporation
3565 Harbor Boulevard
Costa Mesa, California 92626
800.FILENET (345.3638)
Outside the U.S., call:
1.7 1 4 .3 2 7 .3 4 0 0
www.filenet.com

Contents

End User License Agreement (EULA)

License Agreement

READ THIS NOTICE CAREFULLY, THE SOFTWARE IS THE PROPRIETARY INTELLECTUAL PROPERTY OF FILENET CORPORATION AND IS SUBJECT TO THE MINIMUM TERMS AND CONDITIONS SET FORTH BELOW. THESE TERMS AND CONDITIONS MAY BE SUPERCEDED BY THE TERMS AND CONDITIONS OF THE SOFTWARE LICENSE ENTERED INTO BY YOUR EMPLOYER FOR THE USE OF FILENET SOFTWARE. BY USING THE SOFTWARE, YOU ACKNOWLEDGE IT IS FILENET PROPRIETARY INTELLECTUAL PROPERTY AND THAT A VALID SOFTWARE LICENSE WITH FILENET CORPORATION IS APPLICABLE. THEREFORE AT A MINIMUM, YOU AGREE TO BE BOUND BY THE FOLLOWING FILENET END USER SOFTWARE LICENSE TERMS AND CONDITIONS (HEREINAFTER “Agreement”):

1. **Definition of Software**

   The software consists of software owned by FileNet, as well as software owned by certain third party providers (“Third Party Providers”). Each software product includes any documentation relating to or describing such software, such as, logic manuals, flow charts, reference materials, and improvements or updates provided by FileNet (software and documentation collectively called "Software").

2. **Grant of License**

   A. Each Software product, including any documentation relating to or describing such Software, such as, but not limited to, manuals, flow charts and improvements or updates provided by FileNet (collectively "Software"), is furnished to End User under a personal, non-exclusive, nontransferable license solely for End User's own internal use on End User’s servers and client devices (“System”) in compliance with this license and all applicable laws and regulations. End User agrees that this license does not permit End User to: (1) use the Software for a service bureau application or (2) rent, lease, or sublicense the Software; (3) modify or remove any proprietary notices; or (4) transfer the Software without prior written consent from FileNet. The Software is licensed to the End User, not sold.

   B. The Software may only be copied, in whole or in part (with the proper inclusion of FileNet's copyright notice and any other proprietary notice and/or trademarks on such Software), as may be necessary and incidental for archival purposes or to replace a worn or defective copy.

   C. Title to and ownership of the Software and any portions (or any modifications, translations, or derivatives thereof, even if unauthorized) and all applicable rights in patents, copyrights and trade secrets in the Software shall remain exclusively with FileNet and its licensors, if any. Software provided hereunder is valuable, proprietary, and unique, and End User agrees to be bound by and observe the proprietary nature thereof as provided herein. End User agrees to take diligent action to fulfill its obligations hereunder by instruction or agreement with its employees or agents (whose confidentiality obligations shall survive termination of employment or agency) who are permitted access to the Software. Access shall only be given on a need-to-know basis. Except as set forth in this Agreement or as may be permitted in writing by FileNet, End User shall not use, provide or otherwise make available the Software or any part or copies thereof to any third party. End User shall not reverse engineer, decompile or disassemble the Software or any portion thereof, nor otherwise attempt to create or derive the source code. End User acknowledges that unauthorized reproduction, use, or disclosure of the Software or any part thereof may cause irreparable injury to FileNet and/or its licensors, who may therefore be entitled to injunctive relief to enforce these license restrictions, in addition to any other remedies available at law, in equity, or under this Agreement. Further, the trademarks are owned by the respective trademark holder.

   D. FileNet agrees that End User’s affiliates (business entities of which End User owns or controls more than fifty {50%} percent of the voting rights or the controlling body of the business entity) may use the Software; provided that prior to any affiliate’s use of the Software: (i) End User accepts responsibility for the acts or omissions of such affiliates as if they were End User’s acts or omissions; (ii) End User shall indemnify FileNet against losses or damages suffered by

FileNet arising from breach of this Agreement by any such affiliate; and (iii) such use shall not constitute an unauthorized exportation of the Software or documentation under U.S. Government laws and regulations.

3. **Termination.** FileNet shall have the right to terminate End User's license if End User fails to pay any and all required license fees or otherwise fails to comply with these license terms and conditions. Upon expiration of the license term or upon notice of such termination, End User shall immediately return or destroy the Software and all portions and copies thereof as directed by FileNet and, if requested by FileNet, shall certify in writing as to the destruction or return of the same. All confidentiality and non-disclosure obligations herein shall survive termination.

4. **Limited Warranty**

   A. FileNet warrants that it has good and clear title to or has the right to sublicense the Software being licensed hereunder, free and clear of all liens and encumbrances.

   B. FileNet warrants for a period of ninety (90) days from the Shipment Date, Software used in a manner for which it was designed will perform the functions described in the applicable FileNet documentation supplied at the time of delivery provided that, (i) Software is continuously subject to a FileNet Software Support contract, (ii) any substantial nonconformance is reproducible, and (iii) the substantial nonconformance is not caused by third party software or hardware not specified in FileNet's documentation or not expressly authorized in advance by FileNet. FileNet's sole obligation and liability hereunder shall be to use reasonable efforts to remedy any material non-conformance, which is reported to FileNet in writing within the warranty period.

   C. End User accepts sole responsibility for, system configuration, design and requirements, selection of the software for the intended results, modifications, changes or alterations.

   D. THERE ARE NO OTHER WARRANTIES, EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE OR NONINFRINGEMENT WITH RESPECT TO THIS AGREEMENT, THE AGREEMENT, OR THE SOFTWARE LICENSED HEREUNDER. FILENET DOES NOT WARRANT THAT THE OPERATION OF THE SOFTWARE WILL BE UNINTERRUPPTED, THAT THE SOFTWARE IS ERROR-FREE, OR THAT ALL ERRORS CAN BE CORRECTED.

5. **Export.** End User agrees that the Products purchased hereunder will not be exported directly or indirectly, separately or as part of any system, without first obtaining a license from the U.S. Department of Commerce or any other appropriate agency of the U.S. Government, as required.

6. **Restrictions on Use.** End User acknowledges that one of FileNet's Third Party Providers provides application integration software as part of the Software (the "Integration Software"). End User agrees to use the Software solely as follows: (i) FileNet's workflow or content management software will be the triggering source and/or the destination of the transaction managed by the Integration Software; (ii) FileNet's workflow or content management functionality must be a necessary part of the transaction managed by the Integration Software such that without such FileNet functionality, the transaction managed by the Integration Software could not process; (iii) the Software will not be used or configured in such a way as to only provide the functionality that the Integration Software provides; and (iv) all derivative works of the Software made by or for End User are subject to the foregoing restrictions.

7. **Choice of Law.** The laws of the State of California will govern the construction and operation of this Agreement without regard to the conflict of laws provisions thereof.

8. **U.S. Government Restricted Rights**. The Software is Commercial Software and the Software and Documentation are provided with Restricted Rights. Use, duplication or disclosure by the Government is subject to restrictions as set forth in paragraph (c) (1) (ii) of the Rights in Technical Data and Computer Software Clause at DFARS 252.227-7013 or subparagraphs (c) (1) and (2) of the Commercial Computer Software-Restricted Rights at 48 CFR 52.227-19, as applicable. Contractor/manufacturer is FileNet Corporation, 3565 Harbor Blvd., Costa Mesa, California 92626.

Legal Notices

About this Manual

This manual provides the details of the functionalities and features offered by Inbound Document Linker (IDL) 3.2 product.

Education

FileNet provides various forms of instruction. Please visit the Global Learning Services in FileNet's Service & Support area at http://www.filenet.com/

Comments and Suggestions

FileNet invites all customers to communicate with the documentation group on any question or comment related to FileNet manuals and online help. Send an email to docs@filenet.com. We will make every effort to respond within one week. Your suggestions help us improve the products we deliver.

1
IDL 3.2 an Overview

Inbound Document Linker (IDL) enables batch linking of FileNet documents with Siebel records. The rules are defined to specify the match criteria. Based on matching rules defined in the Link Rules table of the Application Connector for Siebel 7 (ACS7) database, IDL matches the properties of a FileNet document to the properties of Siebel records. If a FileNet document and a Siebel record match, a link record is inserted into the Document Link table.

The user can use a FileNet Queue or a flat file containing FileNet document identifiers as input to the Inbound Document Linker. The FileNet Queue can be a FileNet Distributor queue for new FileNet documents or an existing queue that is pre-filled with value by another application.

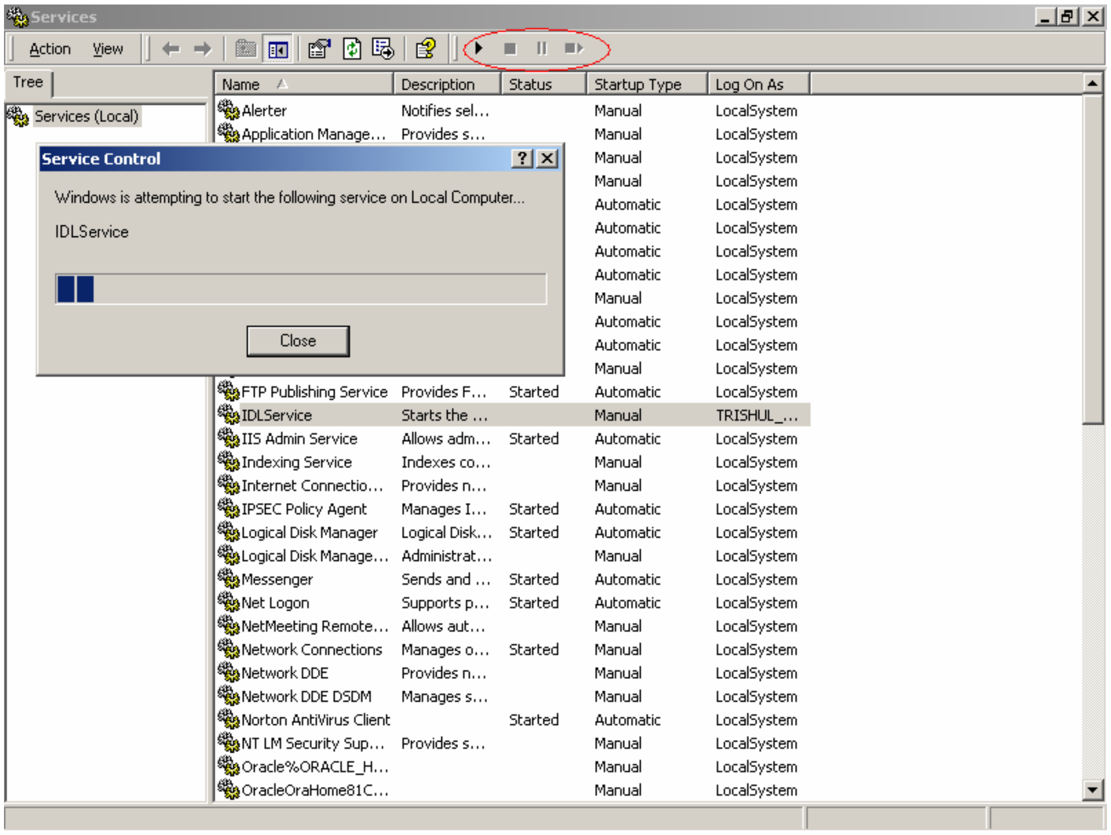
IDL runs as Windows Service on the machine in unattended mode. IDL Service periodically starts batch linking process based on configurable *'Run IDL Every … minute'* parameter. IDL Service can process multiple queues in parallel if queues are configured using *IDL Configuration Tool*. Link activity may be logged, along with any warning or error conditions to local event and error log files. For each input source (in case of queue, it can be multiple), one event log file is created.

2
Running IDL Service

IDL is executed as a Windows service and can be set up to periodically start linking process. To run the service:

1. Navigate to **Start ->Settings->Control Panel->Administrative Tools->Services** to open the Services Control Manager.

2. Select **IDLService** from the list of services and start it. A progress bar appears, as shown below:



Service Control

Windows is attempting to start the following service on Local Computer...

IDLService

Name	Description	Status	Startup Type	Log On As
Alerter	Notifies sel...		Manual	LocalSystem
Application Manage...	Provides s...		Manual	LocalSystem
			Manual	LocalSystem
			Manual	LocalSystem
			Automatic	LocalSystem
			Automatic	LocalSystem
			Automatic	LocalSystem
			Automatic	LocalSystem
			Manual	LocalSystem
			Automatic	LocalSystem
			Automatic	LocalSystem
			Manual	LocalSystem
FTP Publishing Service	Provides F...	Started	Automatic	LocalSystem
IDLService	Starts the ...		Manual	TRISHUL_...
IIS Admin Service	Allows adm...	Started	Automatic	LocalSystem
Indexing Service	Indexes co...		Manual	LocalSystem
Internet Connectio...	Provides n...		Manual	LocalSystem
IPSEC Policy Agent	Manages I...	Started	Automatic	LocalSystem
Logical Disk Manager	Logical Disk...	Started	Automatic	LocalSystem
Logical Disk Manage...	Administrat...		Manual	LocalSystem
Messenger	Sends and ...	Started	Automatic	LocalSystem
Net Logon	Supports p...	Started	Automatic	LocalSystem
NetMeeting Remote...	Allows aut...		Manual	LocalSystem
Network Connections	Manages o...	Started	Manual	LocalSystem
Network DDE	Provides n...		Manual	LocalSystem
Network DDE DSDM	Manages s...		Manual	LocalSystem
Norton AntiVirus Client		Started	Automatic	LocalSystem
NT LM Security Sup...	Provides s...		Manual	LocalSystem
Oracle%ORACLE_H...			Manual	LocalSystem
OracleOraHome81C...			Manual	LocalSystem

3
Viewing IDL Event and Error Logs

To view the Event log and Error logs:

1. Open the directory specified in the Event Log Path and Error Log Path of *IDL 3.2 Configuration Tool*.

2. Open the required log file.

The user should always set Event Log and Error Log parameters to true in *IDL 3.2 Configuration Tool* to enable logging of all activities, warnings and errors. Each run creates a new event log, with timestamp. The error log file is created only if an error occurs. The format for event log is:

IDL_Instance_*n*_Event_Log_*yyyymmdd_hhmmss*.log

where *yyyymmdd_hhmmss* is the timestamp.

Error log files are named as IDL_Error_Log_*yyyymmdd_hhmmss*.log. Sample Event Log and Error log files are given in Appendix A.

It is the responsibility of the system administrator to delete obsolete event and error log files. No automatic deletion is provided.

4
User Exit Application

User Exit application can be a Windows DLL or COM DLL. User Exit application needs to support an entry function by name *'Process'* as follows:

For VB based User Exit application, the signature of 'Process' function is:

```
Public Function Process(
        ByVal p_conIDLDB As ADODB.Connection,
        ByVal p_strDatabaseType As String,
        ByVal p_conSiebel As SiebelDataServer.SiebelApplication,
        ByVal p_strFnLibrary As String,
        ByVal p_strDocClass As String,
        ByVal p_strDocID As String,
        ByVal p_strBusComp As String,
        ByVal p_strRowIDs As String,
        ByVal p_strUserExitData As String) As Boolean
```

For VC++ based User Exit application, the signature of 'Process' function is:

```
  bool  Process (ADODB::_ConnectionPtr  pDBConnecton,
                char* sDatabaseType,
                SiebelDataServer::SiebelApplicationPtr  pSiebelConnection ,
                char* FnLibary,
                char* sDocClass,
                long  lDocID,
                char* sBusComp,
                char* sRowID,
                char* sUserExitData)
```

To configure User Exit Application:

1. Launch the *IDL Configuration Tool*.
2. Click **User Exit Setting** to enable User Exit configuration section.
3. Select 'User Exit DLL Type'.

i. Select 'Normal DLL' option if User Exit DLL is normal Windows DLL. .

ii. Select 'COM DLL' option if User Exit DLL is COM DLL..

4. Enter name of User Exit DLL.

    a Enter fully qualified name of User Exit DLL if selected option is 'Normal DLL'. .

    b Enter Program ID of Class containing 'Process' entry function if selected option is 'COM DLL'.

    Format of Program ID: User Exit DLL name. Class Name

5. If User Exit DLL is ‘COM DLL’ then register User Exit DLL.

5

Sample User Exit

The IDL3.2 release media provides sample User Exit Program. This can be referred to write new User Exit Programs.

Brief Explanation of this COM DLL written in VB is as follows:

Project References

- Microsoft ActiveX Data Object Library 2.6 (msado15.dll).
- Siebel Data BusObject Interfaces (sobjsrv.tlb).
- BulkLnkDLL (BulkLnkDLL.dll)
- IDLInsertDLL (IDLInsertDLL.dll)

Class Description

clsUserExitImpl.cls

This class serves as the main program of the Sample User Exit application. The ‘Process’ function is defined in this class. This is the main method that is called from IDL application. This class also contains more methods for other operations explained below.

Class Name	clsUserExitImpl
Extends /Implements	N.A
Package	N.A
Purpose / Description	This is the main class in the Sample Application. IDL instantiates this class to implement User Exit functionality.

Member Functions

1. Process

Structure	Function Process (p_conIDLDB As ADODB.Connection,p_strDatabaseType As String, p_conSiebel as SiebelDataServer.ApplicationObject, p_strLibrary as String, p_strDocClass as String, p_strDocID as string, p_strBusComp as String, p_strRowIDs() as String, p_strUserExitData as String) as Boolean
Protection	Public

Level	
Description	This function is the main entry point for the Sample Application. IDL calls this function after passing all the parameters, mentioned below. The function encapsulates the complete functionality of Sample Application.
Parameters	P_conIDLDB - ADODB Connection p_strDatabaseType - Database Type Oracle or MSSQL p_conSiebel - Siebel Connection Handle p_conIDLDB - IDL Database Connection Handle p_strLibrary - Name of FileNet Library. p_strDocClass - Name of FileNet Document Class. p_strDocID - Document ID. p_strBusComp - Name of Siebel Business Component p_strRowIDs() - Array of matching Siebel Row Ids. p_strUserExitData - User Exit specific data (Not used in current Sample Application)
Return Values	Boolean
Logic	Create object of clsGEL class of BulkLnkDLL. Store value of passed parameters to appropriate variables. Read values from ini file by calling ReadIni method. If ReadIni returns "False", 1. Write in Error Log. 2. Return True. Set oApp object of clsGEL object to Siebel Connection Handle. Call CreateAndLink method to create record in Siebel business component and create link between newly created record and document being processed. Return True.

2. ReadIni

Structure	Function ReadIni () as Boolean
Protection Level	Private
Description	This method reads BulkLink_GEL.ini file and sets the variables of sample application.
Parameters	
Return Values	Boolean

Logic	Read BulkLink_GEL.ini file. Set local variables of the applications to values read from BulkLink_GEL.ini file.

Note: BulkLink_GEL.ini is used by Sample User Exit application that is placed in System folder. (For example “C:\WINNT”)

3.CreateAndLink

Structure	Function CreateAndLink ()
Protection Level	Private
Description	This function creates a record in Siebel business component. The name of this Business Component is passed as parameter from IDL. This function creates a link between newly created record in Siebel and FileNet document, which is currently being processed.
Parameters	
Return Values	
Logic	Create a record in Siebel business component by calling CreateSRRecord() method of clsGEL object. If successfully created • Take RowID of newly created Siebel record. • Create an object of CDLTRecord of IDLInsertDLT. • Set values of member variables of CDLTRecord object. • Create link between newly created Siebel record and FileNet document by calling InsertRecordInDLT method of IDLInsertDLT.dll. If InsertRecordInDLT does not return 0 Write to Error Log that insert failed. Else Write to Error Log that insert is successful.

Module Description

modHelper.bas

This module is provides a helping hand to clsUserExitImpl class and contains all the global variables and common methods.

The module contains all the global variables and common methods.

Member Variables

S.No	Class Variable Name	Type	Description
1	G_intNumCreatableBCs	Integer	This variable stores number of creatable business components.
2	G_strCreatableBCNames()	String	Array that stores names of all creatable business components.
3	G_strBCLinkRuleIDs()	String	Array that stores Link rule ID's for the Creatable business components.
4	G_intNoFNLinkFields()	Integer	Array that stores number of Link Fields per Creatable business components.
5	G_strFNLinkFields()	String	Array that stores Link Fields for the business components.
6	G_intNoBCUpdateFields()	Integer	Array that stores number of business components Update fields for each business component.
7	G_strBCFieldInfo()	udtBCUpdate Fields	Array that stores FN-Siebel pair information for business component.
8	G_intNumNoMatchFields	Integer	This variable stores number of fields of NoMatchRule business component.
9	G_strNoMatchFieldNames( )	String	Array that stores names of Siebel fields of NoMatchRule business component.
10	G_strNoMatchFieldValues()	String	Array that stores values of all Siebel Fields of NoMatchRule business component.
11	G_intNoParentRowIds	Integer	This variable stores number of fields that store RowID of the parent business component as value.
12	G_strParentRowIdNames()	String	Array that contains names of the fields that store RowID of the parent business component as value.

13	G_strNoMatchBCName	String	This variable stores name of the NoMatchRules business component.
14	G_strDocID	String	Stores currently processed Document ID.
15	G_strRowIDs()	String	Array that contains matching Row ID's in Siebel against currently processed Doc ID.
16	G_strLibrary	String	Stores name of the FileNet library currently being processed.
17	G_strDocClass	String	Stores name of the FileNet document class currently being processed.
18	G_strBusComp	String	Stores name of the Siebel business component.
19	G_strUserExitData	String	Stores additional User Exit data passed from IDL.
20	G_conSiebel	SiebelDataServer.ApplicationObject	Stores reference to the Siebel application connection currently used by IDL.
21	G_conIDLDB	ADODB.Connection	Stores reference to the IDL database connection currently used by IDL.

Logging

Error Logging

In case, Sample Application encounters some errors then one Error Log file is created and details of these errors are written to this file.

Name of Log file: *SampleUserExit_YYYYMMDD_HHMMSS.log*.

where,

- YYYY: Current Year
- MM: Current Month
- DD: Current Date
- HH: Current Hour
- MM: Current Minute
- SS: Current Second

This file is created in the same directory where sample application is residing.

6
DB Access

DB Access is a COM DLL that exposes methods used by IDL DB Configuration Tool to interact with IDL database. These methods can be used by custom applications to insert, update and delete dynamic tables and their data in IDL database.

Brief explanation of this COM DLL written in VB is as follow:

Project References

- Microsoft ActiveX Data Object Library 2.7 (msado15.dll).

Class Description

DBAccess

This class serves as the main program of the DBAccess COM DLL. This class contains various methods for the aforementioned database operations.

Class Structure	DBAccess
Extends /Implements	N.A
Package	N.A
Purpose / Description	This is the main class in the DBAccess COM DLL. This class is instantiated by IDL DB Configuration Tool to interact with IDL database.

Member Functions

1. FetchConfiguredLibraries

Structure	Public Function FetchConfiguredLibraries() As Recordset
Protection Level	Public
Description	This method returns all the configured libraries as a recordset.
Parameters	None
Return Values	Recordset

Example	`Dim l_objDBAccess As DBAccess.clsDBAccess`
	`Dim l_objRS As ADODB.Recordset`
	`Set l_objRS = g_objDBAccess.FetchConfiguredLibraries()`
	‘Loop through the resordset to get all the configured libraries
	‘In this example, adding them to a combo box called cmbLibrary
	`        While Not l_objRS.EOF`
	`            cmbLibrary.AddItem l_objRS.Fields(0)`
	`            l_objRS.MoveNext`
	`        Wend`

2. FetchColumnNames

Structure	Public Function FetchColumnNames(p_strTableName As String) As Recordset
Protection Level	Public
Description	This method returns the column names of a table as recordset.
Parameters	p_strTableName - Table Name - String
Return Values	Recordset

Example	`Dim l_objDBAccess As DBAccess.clsDBAccess`
	`Dim l_objRsColumns As ADODB.Recordset`
	'Getting the column names in a recordset
	`Set l_objRsColumns = l_objDBAccess.FetchColumnNames(m_strTableName)`
	`        While Not l_objRsColumns.EOF`
	            'Loop through the recordset to get the column names
	`            l_objRsColumns.MoveNext`
	`        Wend`

3. FetchConfiguredDocClasses

Structure	Public Function FetchConfiguredDocClasses(ByVal p_strLibrary As String) As Recordset
Protection Level	Public
Description	This method fetches all the configured document classes
Parameters	p_strLibrary - Library Name - String
Return Values	Recordset

Example	`Dim l_objDBAccess As DBAccess.clsDBAccess`
	`Dim l_objRS As ADODB.Recordset`
	`Dim l_strLibName as String`
	`l_strLibName = “csdb”`
	`Set l_objRS =`
	`l_objDBAccess.FetchConfiguredDocClasses(l_strLibName)`
	`While Not l_objRS.EOF`
	  'Loop through the recordset to get the doc class names
	  `l_objRS.MoveNext`
	`Wend`

4. FetchConfiguredFields

Structure	Public Function FetchConfiguredFields(ByVal p_strLibrary As String, ByVal p_strDocClass As String) As Recordset
Protection Level	Public
Description	This method fetches all the configured fields for a particular configured document class
Parameters	p_strLibrary - Library Name - String p_strDocClass - Doc Class Name - String
Return Values	Recordset

Example	`Dim l_objDBAccess As DBAccess.clsDBAccess`
	`Dim l_objRsColumns As ADODB.Recordset`
	`Dim l_strLibName as String`
	`Dim l_strDocClassName as String`
	`l_strLibName = “csdb”`
	`l_strDocClassName = “Account”`
	`Set l_objRsColumns =`
	`g_objDBAccess.FetchConfiguredFields(l_strLibName,`
	`l_strDocClassName)`
	`While Not l_objRsColumns.EOF`
	  'Loop through the recordset to get the configured field names
	  `l_objRsColumns.MoveNext`
	`Wend`

5. AddFieldToSource

Structure	Public Function AddFieldToSource(ByVal p_strLibrary As String, ByVal p_strDocClass As String, ByVal p_strField As String) As Boolean
Protection Level	Public
Description	This method fetches the name of the dynamic table corresponding to the passed document class and library name from FN_IDL_TABLESOURCE table of IDL database. After fetching, it inserts the passed field name as a column of the dynamic table.
Parameters	p_strLibrary - Library Name - String p_strDocClass - Doc Class Name - String p_strField - Field Name - String
Return Values	Boolean. True, if the operation is successful and False otherwise.

Example	`Dim l_objDBAccess As DBAccess.clsDBAccess`
	`Dim l_strLibName as String`
	`Dim l_strDocClassName as String`
	`Dim l_strFieldName as String`
	`l_strLibName = “csdb”`
	`l_strDocClassName = “Account”`
	`l_strFieldName = “Account”`
	`If Not g_objDBAccess.AddFieldToSource(l_strLibName, l_strDocClassName, l_strFieldName) Then`
	`      Exit Sub`
	`End If`

6. DeleteFieldFromSource

Structure	Public Sub DeleteFieldFromSource(ByVal p_strLibrary As String, ByVal p_strDocClass As String, ByVal p_strField As String)
Protection Level	Public
Description	This method deletes the column, passed as parameter from the dynamic table of IDL database.

Parameters	p_strLibrary - Library Name - String p_strDocClass - Doc Class Name - String p_strField - Field Name - String
Return Values	None

Example	`Dim l_objDBAccess As DBAccess.clsDBAccess`
	`Dim l_strLibName as String`
	`Dim l_strDocClassName as String`
	`Dim l_strFieldName as String`
	`l_strLibName = “csdb”`
	`l_strDocClassName = “Account”`
	`l_strFieldName = “Account”`
	`l_objDBAccess.DeleteFieldFromSource (l_strLibName, l_strDocClassName, l_strFieldName)`

7. FetchConfiguredSources

Structure	Public Function FetchConfiguredSources() As Recordset
Protection Level	Public
Description	This method fetches the library name, document class name and the dynamic table name from the FN_IDL_TABLESOURCE table in IDL database.
Parameters	None
Return Values	Recordset

Example	`Dim l_objDBAccess As DBAccess.clsDBAccess`
	`Dim l_objRsColumns As New ADODB.Recordset`
	
	`Set l_objRsColumns = l_objDBAccess.FetchConfiguredSources`

8. InsertDocIDInDatabase

Structure	Public Sub InsertDocIDInDatabase(ByVal p_strDocID As String)
Protection Level	Public
Description	This method inserts the document ID passed in the FN_IDL_CSDOCIDLIST

	table of IDL database. This is if table is not configured for document class
Parameters	p_strDocID - Document ID - String
Return Values	None

Example	`Dim l_objDBAccess As DBAccess.clsDBAccess`
	`Dim l_strDocID as String`
	`l_strDocID = "123"`
	
	`l_objDBAccess.InsertDocIDInDatabase (l_strDocID)`

9. DeleteTableFromDatabase

Structure	Public Sub DeleteTableFromDatabase(ByVal p_strLibrary As String, ByVal p_strDocClass As String)
Protection Level	Public
Description	This method deletes the dynamic table from IDL database for the passed library name and document class name.
Parameters	p_strLibrary - Library Name - String p_strDocID - Document ID - String
Return Values	None

Example	`Dim l_objDBAccess As DBAccess.clsDBAccess`
	`Dim l_strLibName as String`
	`Dim l_strDocClassName as String`
	`l_strLibName = "csdb"`
	`l_strDocClassName = "Account"`
	`l_objDBAccess.DeleteTableFromDatabase(l_strLibName, l_strDocClassName)`

10. FetchDocIDsFromDatabase

Structure	Public Function FetchDocIDsFromDatabase() As Recordset
Protection Level	Public
Description	This method fetches all the Document IDs from FN_IDL_CSDOCIDLIST table of IDL database.
Parameters	None

Return Values	Recordset

Example	`Dim l_objDBAccess As DBAccess.clsDBAccess`
	`Dim l_objRsColumns As New ADODB.Recordset`
	`Set l_objRsColumns =`
	`l_objDBAccess.FetchDocIDsFromDatabase`
	`While Not l_objRsColumns.EOF`
	'Loop through the recordset to get the document Ids.
	`l_objRsColumns.MoveNext`
	`Wend`

11. DeleteDocIDFromDatabase

Structure	Public Sub DeleteDocIDFromDatabase(ByVal p_strDocID As String)
Protection Level	Public
Description	This method deletes the document Id passed as parameter from FN_IDL_CSDOCIDLIST table of IDL database.
Parameters	p_strDocID - Document ID - String
Return Values	None

Example	`Dim l_objDBAccess As DBAccess.clsDBAccess`
	`Dim l_strDocID As String`
	`l_strDocID = "123"`
	`g_objDBAccess.DeleteDocIDFromDatabase(l_strDocID)`

12. UpdateDocIDInDatabase

Structure	Public Sub UpdateDocIDInDatabase(ByVal p_strNewDocID As String, ByVal p_strOldDocID As String)
Protection Level	Public
Description	This method updates the document ID in FN_IDL_CSDOCIDLIST table of IDL database.
Parameters	p_strNewDocID - New Document ID - String p_strOldDocID - New Document ID - String
Return Values	None

Example	`Dim l_objDBAccess As DBAccess.clsDBAccess`
	`Dim l_strNewDocID As String`
	`Dim l_strOldDocID As String`
	`l_strNewDocID = “456”`
	`l_strOldDocID = “123”`
	`l_objDBAccess.UpdateDocIDInDatabase(l_strNewDocID, l_strOldDocID)`

13. IsDBConnected

Structure	Public Function IsDBConnected() As Boolean
Protection Level	Public
Description	This method returns a Boolean, which signifies whether the connection to the database exists or not.
Parameters	None
Return Values	Boolean

Example	`Dim l_objDBAccess As DBAccess.clsDBAccess`
	`If l_objDBAccess.IsDBConnected = False Then`
	`    Exit Sub`
	`End If`

14. FetchFromDatabase

Structure	Public Function FetchFromDatabase(ByVal p_strSQL As String) As Recordset
Protection Level	Public
Description	This method executes a select query on the database and returns the results as a recordset.
Parameters	p_strSQL - SQL Query - String
Return Values	Recordset

Example	`Dim l_objDBAccess As DBAccess.clsDBAccess`
	`Dim l_objRsTemp As New ADODB.Recordset`

	`Dim l_strSQL As String`
	`l_strSQL = "Select * from FN_IDL_CSDOCIDLIST where DOCID = ‘123’”`
	`Set l_objRsTemp = l_objDBAccess.FetchFromDatabase(l_strSQL)`
	`While Not l_objRsTemp.EOF`
	    'Loop through the recordset to browse through the results of the ‘SQL Query
	    `l_objRsTemp.MoveNext`
	`Wend`

15. ExecuteSQLQuery

Structure	Public Sub ExecuteSQLQuery(ByVal p_strSQL As String)
Protection Level	Public
Description	This method executes a query on the database, usually INSERT, UPDATE or DELETE.
Parameters	p_strSQL - SQL Query - String
Return Values	None

Example	`Dim l_objDBAccess As DBAccess.clsDBAccess`
	`Dim l_strSQL As String`
	`l_strSQL = "Delete from FN_IDL_SOURCE1 where DOCID = ‘123’”`
	`Call l_objDBAccess.ExecuteSQLQuery(l_strSQL)`

7
Creating a Link Record

IDL 3.2 provides a helper ActiveX DLL **InsertDLTRecord** to create a link record in Document Link table of ACS7 database. This DLL is used, whenever there is a need to create link record by any third party application. This DLL insulates third party application from IDL database schema. The following are details of this DLL:

Class name: clsInsertDLT

Function Signature: Public Function InsertDLTRecord (

```
ByVal p_IDLDBConnection As ADODB.Connection,
ByVal p_strDatabaseType As String,
ByVal p_strBusComp As String,
ByVal p_strRowID As String,
ByVal p_strLibrary As String,
ByVal p_strDocClass As String,
ByVal p_strDocID As String,
ByVal p_strCreatedBy As String) As Boolean
```

Input Parameters:

Sr. No.	Parameter	Type	Description
1	p_IDLDBConnection	ADODB. Connection	Represents an active connection to IDL database.
2	p_strDatabaseType	String	Represents types of RDBMS used for IDL database. Valid values are: ‘O’: Oracle ‘S’: SQL Server
3	p_strBusComp	String	Represents Siebel business component
4	p_strRowID	String	Represents Siebel row ID to be linked
5	p_strLibrary	String	Represents FileNet library

6	p_strDocClass	String	Represents document class of the DocID
7	p_strDocID	String	Represents document ID
8	p_strCreatedBy	String	Represents an application, which is creating link record.

Output Parameter:

Sr. No.	Type	Description
1	Boolean	If link record creation is successful then the function returns ‘True’ else returns ‘False’.

Appendix A

Sample format of Event Log File

```
2002/09/16 10:13:41,IDL Instance starting Doc ID Processing
2002/09/16 10:13:44,Log on to FileNet Library successful
2002/09/16 10:13:44,Log on to Siebel Application Server successful
2002/09/16 10:13:46,Log on to IDL Database successful
2002/09/16 10:13:46,File Name = "C:\FileNet\IDL\DocIDS.txt", Document
Source File Opened
2002/09/16 10:13:46,Before Fetching DocID from Source File
2002/09/16 10:13:46,After Fetching DocID from Source File
2002/09/16 10:13:46, Before fetching Document class
2002/09/16 10:13:49, After fetching Document class
2002/09/16 10:13:49,Before Fetching Link Rules
2002/09/16 10:13:49,After Fetching Link Rules
2002/09/16 10:13:49,Before Processing Link Rule
2002/09/16 10:13:49,Before Fetching Field Value
2002/09/16 10:13:50,After Fetching Field Value
2002/09/16 10:13:50,Before Executing query to Siebel
2002/09/16 10:13:51,DocID = 196212, Business Component = "Account",
Siebel Field Name = "Name", Matching Siebel Row ID Not Found
2002/09/16 10:13:51,After Executing query to Siebel
2002/09/16 10:13:51,After Processing Link Rule
2002/09/16 10:13:51,Before Processing Link Rule
2002/09/16 10:13:51,Before Fetching Field Value
2002/09/16 10:13:51,After Fetching Field Value
2002/09/16 10:13:51,Before Executing query to Siebel
2002/09/16 10:13:51,DocID = 196212, Business Component = "Account",
Siebel Field Name = "Location", Matching Siebel Row ID Not Found
2002/09/16 10:13:51,After Executing query to Siebel
2002/09/16 10:13:51,After Processing Link Rule
2002/09/16 10:13:51,Before Fetching DocID from Source File
2002/09/16 10:13:51,No More Document ID found in Source File
2002/09/16 10:13:51,Document Source File Closed
2002/09/16 10:13:51,IDL Instance completed Doc ID Processing
```

Sample format of Error Log File

```
Time: 2002/09/25 10:24:09
Error Description: Error occurred while fetching link rules from
'LinkRulesTable'
Error Code: 3127
Error Message: IDispatch error #3127
```