

Module I

Introduction to

Enterprise Toolkit

Solutions™



333 Earle Ovington Blvd

Mitchel Field, New York 11553

Tel: (516) 227-6600

Fax: (516) 394-1196

Website: <http://www.olf.com>

Revision 1.3 Updated Dec 11, 2001

CONTENTS

- 1. BEFORE WE BEGIN..... 5**
 - 1.1 Prerequisites for This Course.....5**
 - 1.2 Recommended Configuration for Workstations5**
 - 1.3 Notes for the Instructor..... 6**
 - 1.4 Before We Start 6**
- 2. OBJECTIVES OF THIS MODULE..... 7**
- 3. TOOLKIT CAPABILITIES 8**
- 4. DELIVERABLES..... 9**
 - 4.1 Enterprise TK Standard Package9**
 - 4.1.1 The API in Enterprise TK Standard DLL Files.....9
 - 4.1.2 Libraries Included and VB Modules9
 - 4.2 Enterprise TK Pro..... 10**
 - 4.2.1 Enterprise TK Pro API 11
 - 4.2.2 Libraries Included and VB Modules 11
 - 4.3 File Locations..... 14**
 - 4.4 Documentation 16**
 - 4.5 Toolkit Programming Resources..... 17**
 - 4.5.1 Navigating the TK Glossary of Functions..... 18
 - 4.5.2 Navigating the Endur/Findur Data Model 20
 - 4.5.3 More Details About the Data Model..... 21
- 5. DESCRIPTION OF TOOLKIT 23**
 - 5.1 Architecture..... 23**
 - 5.2 Toolkit Functionality 24**
 - 5.3 Toolkit Configuration to Run Applications 27**

- 5.3.1 The Run Scripts 27
- 5.3.2 Licenses Required..... 28
- 6. OPERATIONAL MODES 29**
 - 6.1 The Attach Mode of Operation..... 29**
 - 6.1.1 What Is Attach Mode? 29
 - 6.1.2 Why Run in Attach Mode?..... 29
 - 6.1.3 The OlfapiAttach.ocx Object..... 30
 - 6.1.4 The Attach Method of the OlfapiAttach.ocx 31
 - 6.1.5 Developing and Running Attach Mode Applications..... 32
 - 6.1.6 Example 1: Attach to Endur/Findur 36
 - 6.1.7 Example 1 Code..... 37
 - 6.1.8 Creating the OlfapiAttach.ocx Versus Dropping the Object 39
 - 6.2 Stand-alone Mode 40**
 - 6.2.1 What is Stand-alone Mode? 40
 - 6.2.2 Why Run in Stand-alone Mode?..... 40
 - 6.2.3 The Stand-alone Method 41
 - 6.2.4 Example of TK_Init Code 42
- 7. EXERCISES 49**
- APPENDIX I – TYPICAL RUNSCRIPT FILE 50**

Symbols Used



Note:
Additional information.



Important:

Important additional information



Tip:

An alternative method to perform the same function.

1. BEFORE WE BEGIN

Important Windows note about your mouse settings

To be sure that instructions work exactly as given, please go to Start, Control Panel and click on the mouse settings button. Select the following:

Single click to open an item, point to select.

1.1 Prerequisites for This Course

- Visual Basic 6.0 Programming Experience
- Understanding of ActiveX technology in Visual Basic
- Understanding relational database concepts
- Endur/Findur foundation training or equivalent experience
- Familiarity with the commands used in the Windows command prompt. For more information, type **cmd** and then **Help** from the **cmd** prompt.

1.2 Recommended Configuration for Workstations

The following minimum hardware and software are required for Visual Basic applications working with Endur/Findur:

- Microsoft Windows NT 4.0 or Windows 2000
- For this course, you will need Microsoft Internet Explorer 5.0 or later to browse on-line reference
- Endur/Findur availability and connectivity
- Visual Basic 6.0 for developers and Visual Studio Service Pack III or later updates
- For Module IV - Reporting, which is part of this training course; you will need access to Microsoft Excel and Crystal Reports
- Enterprise Toolkit should have been configured in your machines and configured to work with the latest release of Endur/Findur by the systems administrator.

⚡ Installation of Toolkit is not covered in this module. Your system administrator (SA) needs to install the required software prior to running the examples according to the Toolkit Installation Guide. The SA also needs to make sure there is no more than one OlfapiAttach.ocx registered on the machine, and that registration should be pointing to an existing directory.

1.3 Notes for the Instructor

There is a separate document available that details classroom setup for the instructor.

1.4 Before We Start

As indicated by the instructor, all students are to go to the folder where your Toolkit installation resides and write down the path here:

Path of Toolkit installation.

All students go to the location of the test runscript that you will be using for this class and write down its absolute path here:

Path to test runscript.

Locate the Examples folder in your Toolkit directory and copy it to your local or C: drive (your workstation's hard drive). You will be using these examples during this course and you will be asked to load them into your Visual Basic editor. Unless otherwise instructed, whenever you are asked to locate an example in this course, access the local copy of the Examples folder.

Note that you will also need to reload any standard modules in the project. You will be directed to do so by your instructor before you run any of the examples.

In some classes, students might be instructed to run the examples from the network if the students have the correct security access to the drive.

2. OBJECTIVES OF THIS MODULE

The objectives of this module are to demonstrate:

- The capabilities of the Toolkit product
- The libraries and files included with Standard and Pro versions
- The tools needed by the programmer: glossary, user's guides, and data model documentation
- The modes of operation: Attach and Stand-alone.

3. TOOLKIT CAPABILITIES

The Enterprise Toolkit Solutions™ package is a powerful set of Visual Basic functions that constitute the full Applications Programming Interface (API) to the OpenLink Endur/Findur system. There are two versions of the product, Enterprise Toolkit (TK) Pro™ and Enterprise TK Standard™, referred to as Toolkit throughout the product documentation.¹

Using Toolkit, your in-house programmers can create Visual Basic applications that access the Endur/Findur database to extract information, manipulate the data, and add information in real time in order to create customized extensions to Endur/Findur. The predefined functions provided also help you to generate reports and manipulate data.

With Toolkit, you can create complete Visual Basic stand-alone applications or you can create Endur or Findur-dependent applications that run seamlessly alongside (attached) the Endur/Findur system.

Here are some typical applications that can be created with Toolkit:

- Graphical user interfaces (GUIs)
- Custom reports
- Analysis tools
- Deal entry screens
- Curve updating screens
- Access to other databases on your system
- Internal Toolkit developed solutions
- Other custom processing.

¹ Throughout these training documents, Enterprise Toolkit Solutions will be referred to as Toolkit
© 2001 OpenLink

4. DELIVERABLES

Toolkit is available in two editions: Enterprise TK Standard and Enterprise TK Pro. The following are descriptions of the items delivered in your Toolkit CDs.

4.1 Enterprise TK Standard Package

Enterprise TK Standard is used for developing custom reports and outbound interfaces. Visual Basic applications created with Enterprise TK Standard run alongside a running session of Endur/Findur so that users can switch from the VB application to Endur/Findur. This is called the "Attached" mode (more details below).


Typical applications include:

- Reporting
- Data transfer to applications such as Excel
- Dynamic updating of reports and graphs.

4.1.1 The API in Enterprise TK Standard DLL Files

Enterprise TK Standard comes with a complete API (Applications Programming Interface) for accessing any stored data and for writing and generating reports. These libraries are accessible through the following DLL (dynamic linked libraries) files, which are included with Enterprise TK Standard.

- olfapi_reporting.dll

 *The DLL files must be installed on your network or the Toolkit will not run.*

What Is a Module?

The .bas files shipped are called *modules* in Visual Basic terminology. Each of the modules listed in the table below contains a list of Declare statements, as shown on page 13, which make it easier to code your Visual Basic application. You load modules to Visual Basic from the **Project menu** by clicking **Add Module**.



If you are utilizing only a limited number of functions in your application and do not want to load the Toolkit modules, you can copy the Declare statements from the module directly into your code. This might save memory and be more efficient.

4.1.2 Libraries Included and VB Modules

Table 1 - 1 Enterprise TK Standard DLL - olfapi_reporting.dll, shown below, lists the function libraries typically included with the Standard version as well as the Visual Basic module files (with .bas extension) that contain the prototypes for Toolkit functions, enumerations, and constants.



The .bas files are called standard modules in Visual Basic terminology. Each of the modules listed below contains a list of Declare statements that make it easier to code

your Visual Basic application. You load standard modules to Visual Basic from the Project menu by clicking Add Module. Alternatively, you can copy and paste the Prototype Declare statement from the Module (see 4.5.1). More information on this topic is available in Module III of this training program.

Table 1 - 1 Enterprise TK Standard DLL - olfapi_reporting.dll

Library name	VB module(s)
Crystal Report Library	olfapi_crystal.bas
Database Library	olfapi_db_table.bas
Date Parsing Library	olfapi_date.bas
Math Library	olfapi_math.bas
Query Library	olfapi_query.bas
AVS Reporting Library	olfapi_avs_report.bas
Simulation Library	olfapi_reval.bas
String Library	olfapi_str.bas
High Table Library	olfapi_high_table.bas
Low Table Library	olfapi_low_table.bas
Utility Library	olfapi_util.bas
Constants Library	olfapi_const.bas
Reporting Library	Olfapi_reporting.bas

4.2 Enterprise TK Pro

Enterprise TK Pro extends the capabilities of Enterprise TK Standard by including additional functions that allow development of complete stand-alone applications that run independently of Endur/Findur. Users can build custom data entry screens and related processes that work seamlessly with the core OpenLink system as well as interfaces in Visual Basic, Excel, or Access, etc.

Typical applications that can be created with Enterprise TK Pro include:

- Trade booking
- Market data curve manipulation
- Custom Graphical User (GUI) Interfaces

4.2.1 Enterprise TK Pro API

Enterprise TK Pro comes with a complete API (Applications Programming Interface) for accessing any stored data and for writing and generating reports. The Enterprise TK Pro package includes these DLLs.

- olfapi_reporting.dll (also included in TK Standard)
- olfapi.dll
- olfapi_adv_reporting.dll

4.2.2 Libraries Included and VB Modules

Table 1 - 2 Libraries in Enterprise TK Pro, lists the libraries typically included with the Enterprise TK Pro version as well as the Visual Basic standard module files (with extensions .bas) that contain the prototypes for Toolkit functions, enumerations, and constants. The modules are located in the **olfapi** folder.

Table 1 - 2 Libraries in Enterprise TK Pro

Library name	VB module(s)
The files below are available with TK Standard and TK Pro:	
Crystal Report Library	olfapi_crystal.bas
Database Library	olfapi_db_table.bas
Date Parsing Library	olfapi_date.bas
Math Library	olfapi_math.bas
Query Library	olfapi_query.bas
Reporting Library	olfapi_avs_report.bas
Simulation Library	olfapi_reval.bas
String Library	olfapi_str.bas
Table Library	olfapi_high_table.bas
Table Library	olfapi_low_table.bas
Utility Library	olfapi_util.bas
Constants Library	olfapi_const.bas
Reporting Library	olfapi_reporting.bas
The files below are available only with TK Pro:	
Accounting Library	olfapi_acct.bas
Power Library	olfapi_power.bas
AFS Library	olfapi_afs.bas
Reference Library	olfapi_ref.bas
Broadcast Library	olfapi_broadcast.bas
Services Library	olfapi_services.bas
Index Library	olfapi_idx.bas
Trade Maintenance	olfapi_tmaint.bas
Instrument Library	olfapi_ins.bas
VAR Library	olfapi_var.bas
MQSeries Library	olfapi_mqseries.bas
TRANF Library	olfapi_tranf_api.bas
Database Maintenance	olfapi_db_maint.bas
Volatility Library	olfapi_vol.bas
Advanced Reporting	olfapi_adv_reporting.bas
Professional Reporting	olfapi.bas

A typical .bas standard module with Declare statements

```
Declare Function TABLE_PrintCrystalReport Lib "olfapi_reporting" (ByVal t As Long, ByVal report_name As String) As Long
Declare Function TABLE_PrintCrystalReportWithParams Lib "olfapi_reporting" (ByVal t As Long, ByVal report_name As String, ByVal param_table As Long) As Long
Declare Function TABLE_ViewCrystalReport Lib "olfapi_reporting" (ByVal t As Long, ByVal report_name As String) As Long
Declare Function TABLE_ViewCrystalReportWithParams Lib "olfapi_reporting" (ByVal t As Long, ByVal report_name As String, ByVal param_table As Long) As Long
Declare Function TABLE_ExportCrystalReport Lib "olfapi_reporting" (ByVal t As Long, ByVal report_name As String, ByVal export_type As Long, ByVal export_name As String, ByVal use_outdir As Long) As Long
Declare Function TABLE_ExportCrystalReportWithParams Lib "olfapi_reporting" (ByVal t As Long, ByVal report_name As String, ByVal export_type As Long, ByVal export_name As String, ByVal use_outdir As Long, ByVal param_table As Long) As Long
Declare Function CRYSTAL_PrintReport Lib "olfapi_reporting" (ByVal report_name As String) As Long
Declare Function CRYSTAL_PrintReportWithParams Lib "olfapi_reporting" (ByVal report_name As String, ByVal param_table As Long) As Long
Declare Function CRYSTAL_ViewReport Lib "olfapi_reporting" (ByVal report_name As String) As Long
Declare Function CRYSTAL_ViewReportWithParams Lib "olfapi_reporting" (ByVal report_name As String, ByVal param_table As Long) As Long
Declare Function CRYSTAL_ExportReport Lib "olfapi_reporting" (ByVal report_name As String, ByVal export_type As Long, ByVal export_name As String, ByVal use_outdir As Long) As Long
Declare Function CRYSTAL_ExportReportWithParams Lib "olfapi_reporting" (ByVal report_name As String, ByVal export_type As Long, ByVal export_name As String, ByVal use_outdir As Long, ByVal param_table As Long) As Long
Declare Function CRYSTAL_RunTaskById Lib "olfapi_reporting" (ByVal task_id As Long) As Long
Declare Function CRYSTAL_RunTaskByName Lib "olfapi_reporting" (ByVal task_name As String) As Long
Declare Function CRYSTAL_GetRptDir Lib "olfapi_reporting" () As String
```

4.3 File Locations

The Endur/Findur installation process typically creates the series of directories shown in the diagram below. Toolkit developers will need to be familiar with these files and their locations. Figure 1 - 1 Directory Tree provides a guide to all of the .bas files you will need and their locations.

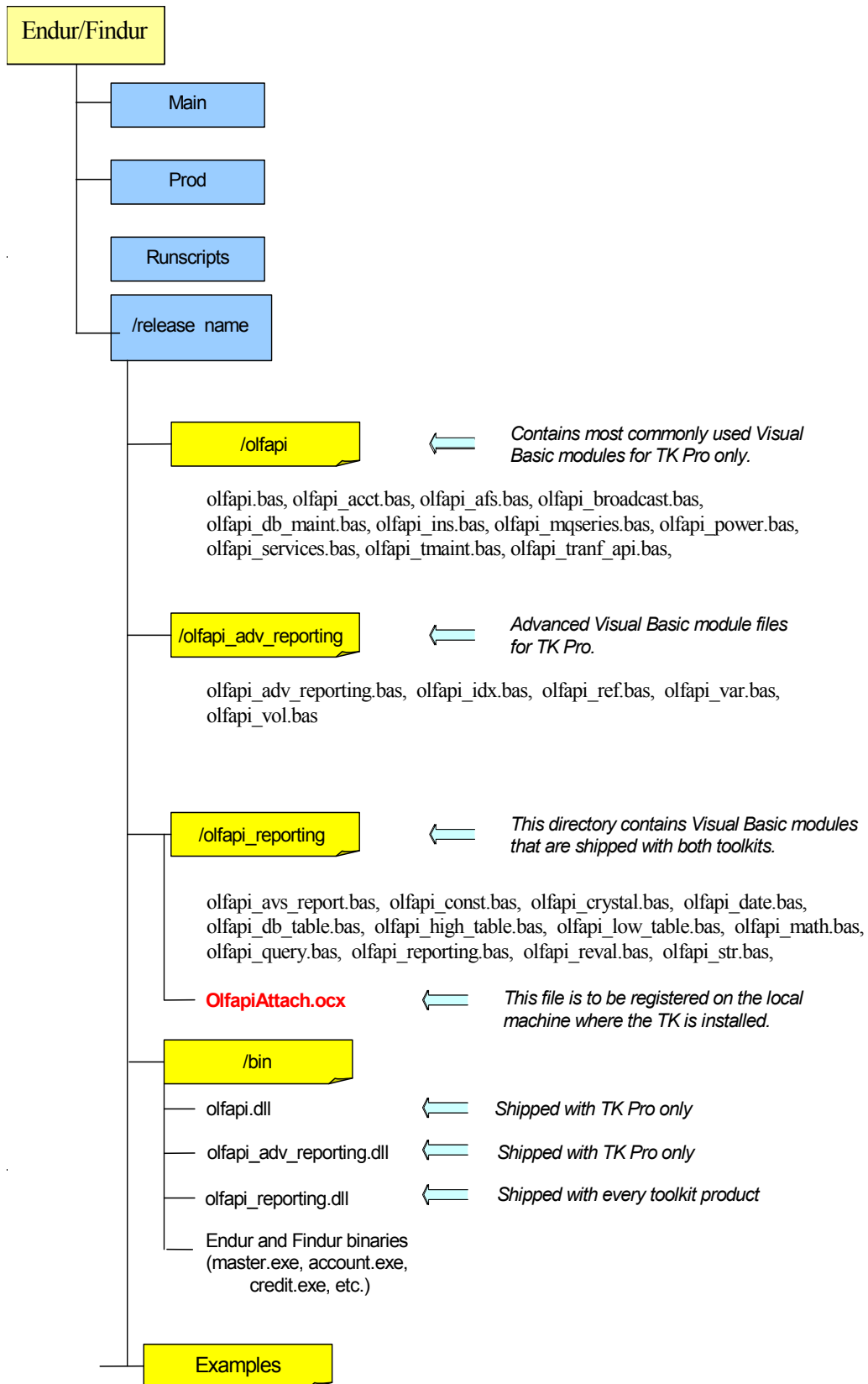


Figure 1 - 1 Directory Tree

4.4 Documentation

The following table lists all of the documentation available for Toolkit. These documents are available on the Toolkit CD.

Table 1 - 3 List of Toolkit Documentation

Name	Business Purpose
VB Glossary	Contains a glossary of Toolkit functions, definitions, and examples.
Data Model	Provides various files describing the Data Model and the ER Studio Viewer needed to analyze the Data Model.
Simulation_Results.chm	User guide describing the System's functionality for generating simulation results.
TK_FAQ.doc	Frequently asked questions (FAQs) about TK; answers provided.
TK_Glossary.chm	A reference source for TK developers. It defines TK functions.
TK_Install.doc	Installation guide with directions on how to: 1) unzip the TK zip file delivered with a client's release, and 2) register the OlfapiAttach.ocx user control.
TK_Known_Issues.doc	A list of either known or potential issues with TK. It provides descriptions of supported platforms and use cases.
TK_Pro_Documentation.doc	Provides an overview of TK Pro and advanced examples using the TK Pro API.
TK_Programmer's_Guide.doc	How to attach with the TK; an intro to the module (.bas) files; how to add TK functions to a VB project.
TK_Std_Documentation.doc	Provides an overview of TK Standard and examples using the TK Standard API.
TK_Training_Workbook.doc	Workbook for TK Standard and TK Pro; how to get the program running and how to create initial TK applications.

4.5 Toolkit Programming Resources

In addition to having good Visual Basic programming skills, Toolkit developers need to be familiar with two important documents:

- The glossary of functions, which is available within the documentation CD. The document describes each function, the format, prototype *declare* statement and an example. The document is presented in a help file and allows navigation through keyword search or by browsing through a tree of categories. The file name is TK_glossary.chm. Instructions on how to navigate the glossary are provided below in section 4.5.1.
- The data model information, which contains all of the information required to utilize tables and fields from Endur/Findur. Programmers will need to know field names and table names for coding certain applications. All of the information you will need on the data model is available in the **Data Model** folder. The data model is viewable using the ER Studio Viewer, provided with the Toolkit software. We will *briefly* discuss data model navigation in section 4.5.2.

☞ We highly recommend that Toolkit programmers review the Data Model document.

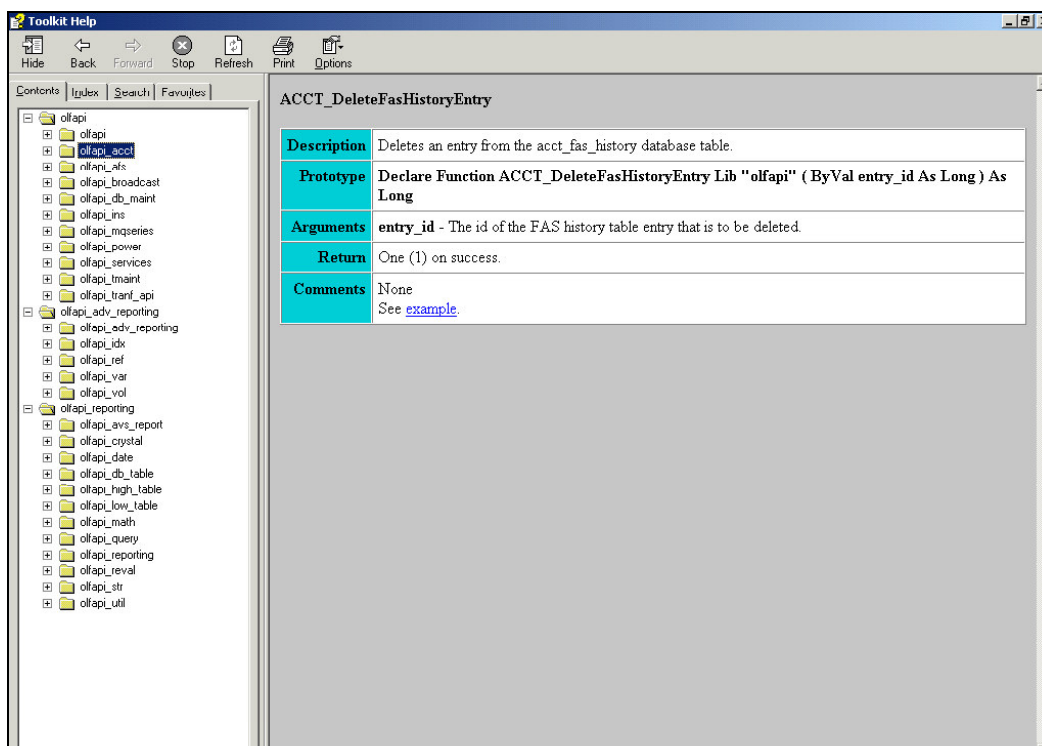
4.5.1 Navigating the TK Glossary of Functions

The following hands-on exercise will familiarize you with the TK_Glossary documentation.



For more details on Toolkit capabilities, spend some time after this course going through functions of interest.

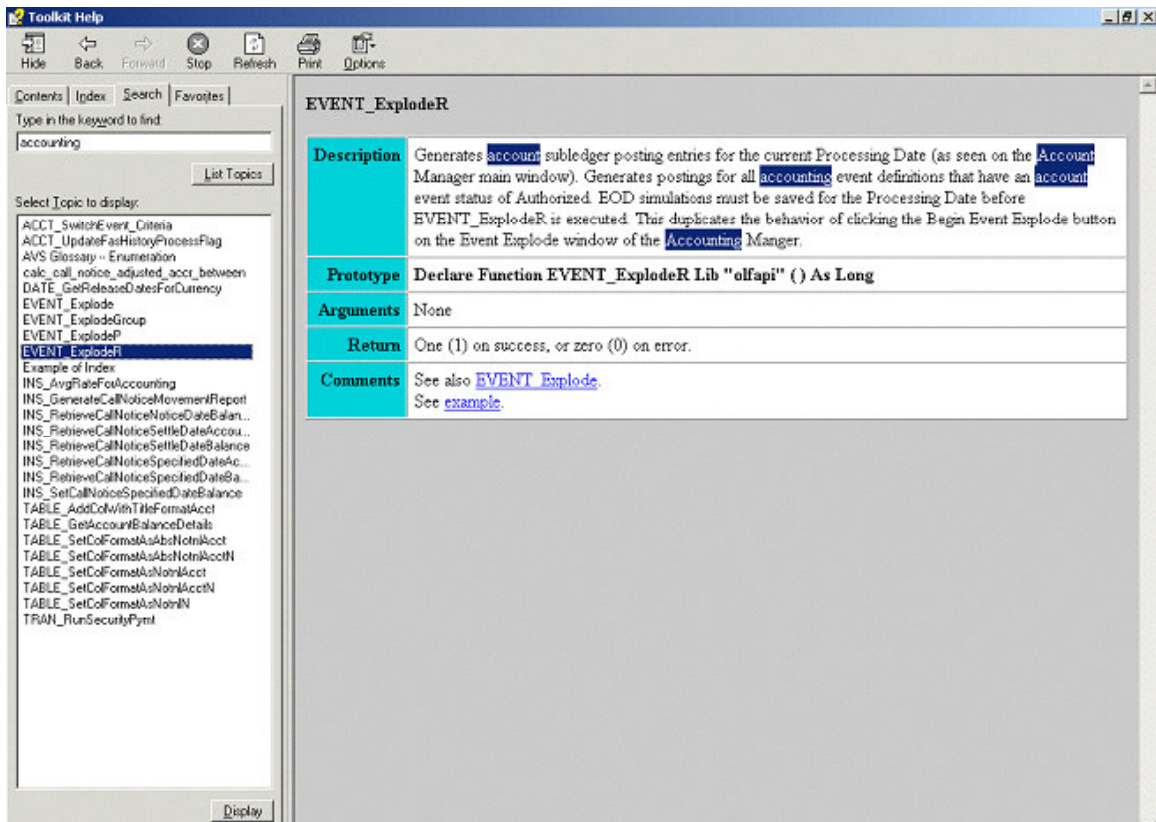
1. Open the file **TK_glossary.chm** contained in the **TK_documentation** folder of the Endur/Findur directory.



2. Browse through the Toolkit Glossary navigator on the left pane. The navigator can be hidden or displayed by using Hide/Show button on the toolbar. Note that most of the directory names correspond to the names of the Visual Basic modules.
3. Click on a function to display it on the right pane. Note that the name of the function is above the description.

🔗 The Prototype box contains the declare statement that needs to be included in your Visual Basic code before you can call the function. This statement is also contained in the corresponding .bas module. The name of the module is the highest level on the tree shown on the left hand side.

4. Click the **Search** tab on the left navigator to display the search box as shown below.



5. Type the keyword **Accounting** into the keyword box and then click the **List Topics** button to display all of the topics that have the keyword Accounting anywhere in the text. Note that the keyword is displayed in reverse print on the right detail window.



You can disable the feature that highlights keywords by selecting Search Highlight from the Options menu in the Glossary toolbar. Try clicking it now and then click on any other function. Notice that keywords are no longer highlighted.

6. For quick retrieval of commonly used functions, click the **Favorites** tab on the navigator. Click the **Add** button to add the selected topic to your list of favorites.
7. Click the **Index** tab to view the topics in alphabetical order.
8. Click the **Example** hyperlink on the topic description to display sample code that utilizes the given function.

4.5.2 Navigating the Endur/Findur Data Model

The purpose of the Enterprise Solutions Data Model and the Enterprise Solutions Data Model Reports documents is to provide information to clients who want to access an Endur/Findur database in order to perform analyses or to develop custom processes, extract data, and generate reports.

The data model and report documents focus on data maintained in the:

- Accounting Manager
- Admin Manager
- Credit/Risk/RTP/Operations Services Modules
- Market Manager
- Operations Manager
- Reference Manager
- Trading Manager

🌀 In order to understand the data model, you will need to have basic OpenLink foundation training and have some knowledge of the business processes.

The following exercise will familiarize you with navigation of the data model. For more details, read the data model documentation included in the Data Model folder on the CD or folder provided with the Endur/Findur system.

- 1.** Click the **Data Model** folder in the Documentation CD.
- 2.** Click the data_model.5.x.dm1 file to display the data model information in the ER viewer. It will take some time to load.
- 3.** Click the first dropdown box and select 100% and in the second dropdown box select data type. Select an item below the sub model to display a screen that will look as follows:

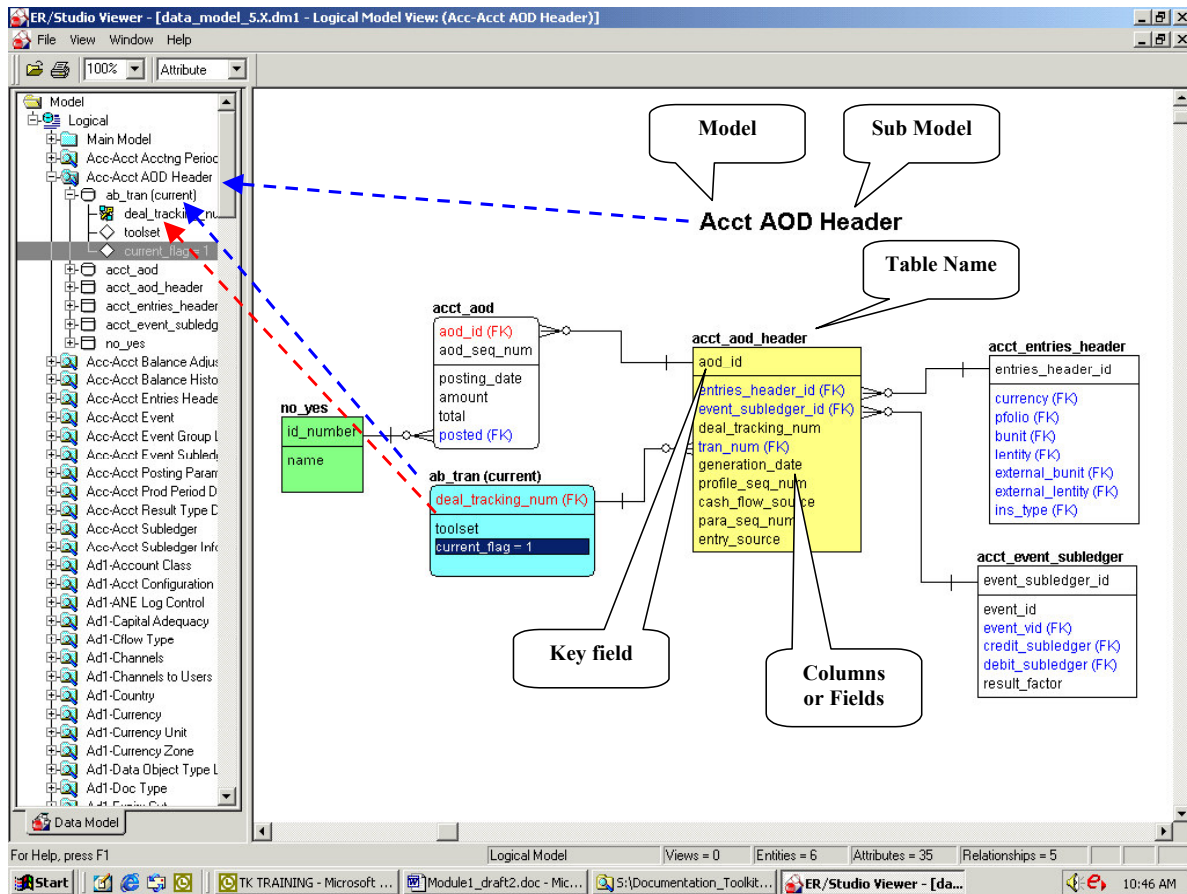


Figure 1 - 2 - Navigating the Data Model

4.5.3 More Details About the Data Model

For more detailed information, refer to the Data Model documentation on the Toolkit CD.

What can I see?

The data model shows all of the tables in the system, the fields and interrelationships graphically. In order to understand what you are looking at, keep in mind the terminology used in the data modeling software is somewhat different than what you are used to:

System Module – is the general category or highest branch. The term corresponds to the categories in Endur/Findur. Each three-letter category prefix in the model corresponds to a system module. For example:

- Acc Account Manager
- Mkt Market Manager
- Ref Reference Manager etc.

The left side of the viewer displays an alphabetized list of all the sub models.

The sub models can be expanded to list the entities they depict. Entities correspond to the tables in the system. Below each table are the field definitions.

You can view an entity definition or a relationship name by placing the cursor on the entity or relationship line.

⚡ It may take a few seconds for the description to appear on the screen.

Data Model Color Conventions

The following color conventions are used throughout the data model.

- | | |
|-------------------|---|
| Yellow | Indicates the focus entity of a sub model or a history entity. |
| Green | Indicates a database table with fixed values that cannot be modified by the user. |
| Light Blue | Indicates a subset of items from a database table. For example, the light blue entity, <code>business_unit (integer)</code> , represents only the internal business units from the business unit table. The entity detail report for a subset entity will always include the SQL required to select the subset of items that it represents. |
| Red | Indicates a database table that is not used in the system. |

How can I customize what I see?

The ER Studio Viewer menu bar contains options for customizing the display features of the model. You can control the zoom level and the notation used for relationships. You can choose any of the following display levels:

- Entities with all their Attributes
- Entities only
- Entities with Primary Keys only
- Entities with Primary and Foreign Keys only
- Entities with all their Attributes and their corresponding data types etc.

5. DESCRIPTION OF TOOLKIT

5.1 Architecture

Toolkit allows applications developed by Microsoft Visual Basic to communicate with the underlying engines and services of Endur/Findur.

Within your Visual Basic applications, you will code calls to Toolkit functions. The Toolkit functions communicate directly with the core functionality of the Endur/Findur engines via .dll files.

The library of over 2000 functions available in Toolkit allows the user to perform numerous operations to create reports, user screens, access databases and create entire stand-alone applications.

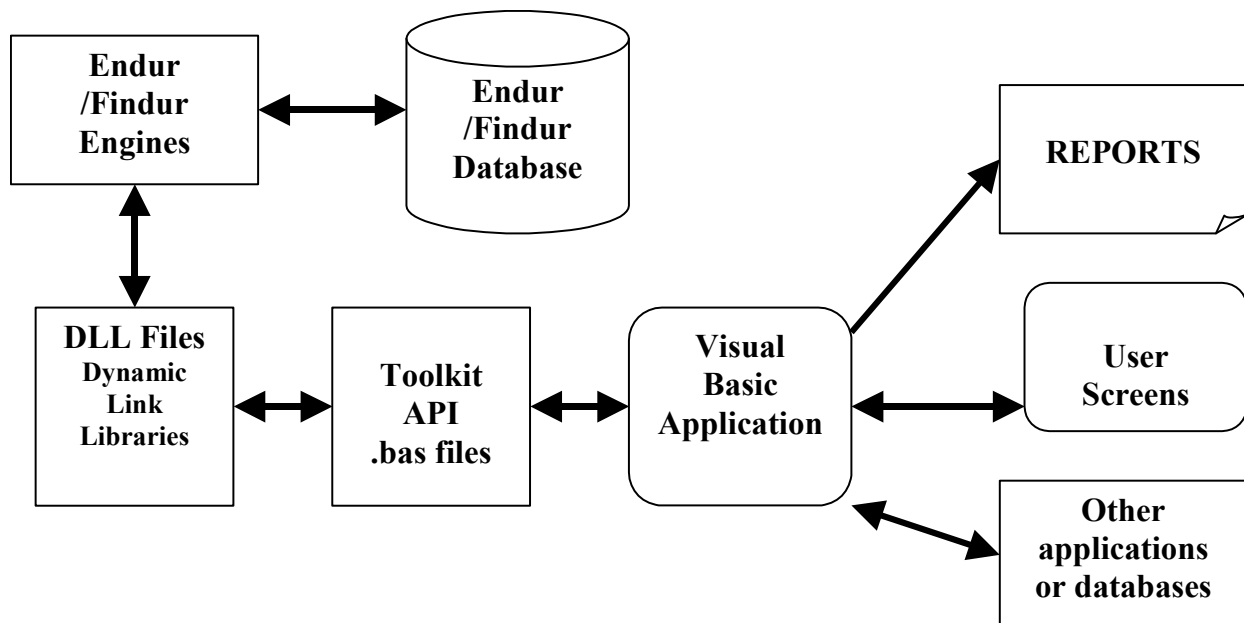


Figure 1 - 3 Software Architecture

5.2 Toolkit Functionality

By now you are familiar with the libraries provided with Toolkit and how to navigate the **Toolkit Glossary** of functions. The following table describes most of the types of functions available and the corresponding prefixes.²

Prefix	Description
ACCT	Accounting Functions associated with End of Day processing.
AFS	Functions used for performing Application File System (AFS) processing
ANE	Processes messages generated from the Automatic Notification Engine (formerly Abacus Notification Engine).
ARCHIVE	Moves data to an archive database table.
BROADCAST	Functions used to send and receive messages to individuals or groups of people contained in a channel.
CLOSEOUT	Functions used for End of Day processing that runs the match off process using various methods.
CREDIT	Enables/Disables monitoring of the given Credit Exposure Definition.
CRYSTAL	Contains all functions associated with the Crystal Reports Reporting Tool. These functions perform actions relevant to the format, generation and manipulation of data reports.
DATE	These functions are used to configure, use, format and manipulate the date and time stamps used throughout the tables.
DBASE	These functions are used to set values for the BCP (Bulk Copy Protocol) for the default database connection of a module or script engine.
DBMAINT	These functions are used to perform database maintenance.
DBTABLE	These functions are commonly used to transfer data between tables in memory to/from the database.
DT	Performs various functions to the Date and Time variables pre-defined in DATE functions.
DW	These functions are used for Data Warehouse processing.
EOD	Functions used for End of Day processing.
EXPORT	These functions export data from a given source file to a destination file using various methods.
HEDGE	These functions are used to perform Hedging functions to transactions.
IMPORT	These functions transfer, control and facilitate data from an import control table to an import destination table. Some functions perform maintenance to the table or the data in the tables.
INDEX	These functions are used to Export or Import market index definitions to/from a text file or an Enterprise Database. They are used to transfer data between databases by converting the data into text files compatible between different databases.
INS	All Instrument functionality is defined with this prefix. Any task performed for the purpose of using, modifying or generating instrument processing is defined using one of these functions.

² Refer to the VB Toolkit Glossary for the latest list of Toolkit functions.

Prefix	Description
MATH	Common functions required for bond and mortgage calculations and data analysis.
MKTD	Performs all Market Data functions used to generate indexes, populate Market data tables and perform Market Data Analysis and maintenance.
MQ	These functions are used to access and perform various functions with the Queue Manager.
OPS	These functions are used to report Operation Services messages. These functions can only be used in Operation Services pre-processing scripts.
ORIEN	ORIEN functions not supported. Intended for use by OpenLink personnel only.
OUTBOUND	These functions are used for database access to set lists of variables or literal values (text or number) to be used during the confirmation/invoicing procedure.
POS	Performs all functions relating to data manipulation of the position page from the RTPE configuration table.
PROCESS	Returns amount of memory utilized
PUBLISH	Used to generate reports which can be published globally or locally.
PWR	These functions are used for all Power Deal transaction and maintenance functionality.
QUERY	These functions are used for the creation, modification, deletion and maintenance of all general Query tasks.
REF	These functions perform lookups and maintenance to Reference Tables associated to an Alias Table definition defined as an enumeration.
RNG	
REPORT	These functions send specified data to a report page to be printed.
RESULT	These functions use result type and result class enumerations to create, modify, delete and maintain result tables containing lists of specified results performed from various calculations in the system.
RISK	Enables/Disables monitoring of the given Risk Exposure Definition. These functions duplicate the behavior of the 'Start/Stop Monitoring' option on the Exposure Definition Menu of an Exposure Definition node in the Risk Manager.
RNG	These functions are used for random number generation.
RTP	These functions define, create, manipulate, retrieve and maintain the Real Time Positioning data on the positioning page created by the Real Time Positioning Engine (RTPE).
RTPE	These functions are used to apply controls to the Real Time Positioning Engine (RTPE).
RTPQUERY	Queries data from an RTP page.
RTPTABLE	Sets and retrieves data from an RTP comm table.
SETTLE	These functions are only for deals that have been amended. Copies transaction event details from the transaction that was amended to the new validated transaction (eg. actual_amount, actual_status, document_id).
SETTLE_INST	These types of functions are used for Settlement Instructions.
SETTLE_SPLIT	These types of functions are used to perform Settlement Splits. Splits a cash settlement event into multiple events.

Prefix	Description
SIM	These functions are used for the Simulation sub-system. Various functions perform configuration and parameter definition. Other functions are used for the input and retrieval of data from the simulation tasks defined.
STR	The STRING functions are used to modify, create, search and return various string values from the specified tables. Many of these functions are utilized for string maintenance. Some functions are used to define a string variable.
SUBSCRIBE	Used to Subscribe to a Published report. Also used to create a subscription table and perform Subscription table functions.
SYSTEM	These functions are used to execute Operating System Commands.
TABLE	Table functions perform all processing available on tables. Table functions are extensive and perform various tasks for the manipulation, input, output, maintenance, format and control of data tables in memory and the database.
TASK	Used to create and run tasks generated from a script.
TIME	Functions used to get the application's database server time represented in various formats.
TK	Toolkit Functions are used to access and utilize Toolkit functionality from other VB applications.
TMAINT	These are transaction maintenance functions used to modify or fix unresolved transaction issues.
TOF	These functions are used for the creation of scripts, which perform Ticket Output Feed (TOF) processing for the OpenLink Reuters 2000 Ticket Feed Interface (TOF interface).
TRAN	Used to define and manipulate every aspect of a transaction in the system.
TRANF	A sub-division of the TRAN category used for transaction
TSERIES	These functions perform Time Series Analysis Configurations, maintenance, data entry and retrieval of information used in the TSERIES computations.
UPGRADE	These functions are used to upgrade previous versions of software and scripts to newer versions. Some post process Upgrade functions can be used after databases are upgraded to newer versions. Note: These functions should not be called from AVS scripts. These are reserved for Toolkit development.
UTIL	The UTIL Prefix is a placeholder for functions created for Utility purposes. These functions perform some Utility function not performed on a regular basis.
VAR	These are the Value at Risk functions used in the Risk Management toolset to define, modify, calculate, maintain all aspect of the VAR definition.
VOL	These functions perform the Volatility processing with the product.

5.3 Toolkit Configuration to Run Applications

Programmers developing with the Toolkit need to understand the initialization process for Endur/Findur in order to run Toolkit and to run Visual Basic applications created with Toolkit.

The initialization file sets the user environment variables, which tell the computer where to find the pieces needed to run the Endur/Findur system and what options are being used in this session. The run script file (written as "runscript" in the system) or files initialize the environment variables.

☞ A table of the environment variables can be found in the Endur/Findur user guide, accessible by clicking Help in any system Manager. Search for the topic "Environment Variables."

5.3.1 The Run Scripts

The run script is a batch file that is used to launch the Endur/Findur system. Most of the environmental variables are set within this file. While generally you will not need to change these, it is important to understand these environment variables, particularly if you are developing complex applications. A typical configuration file is provided in Appendix I. It is annotated with information on the environment variables.

☞ In Windows NT or Windows 2000, it is important to run the Endur/Findur program and VB from the command prompt after these variables are set. User environmental variables will not be known to Toolkit if you switch to Windows and double click on the Endur/Findur or VB icons. For more information about how Windows handles Environment variables, refer to your Microsoft documentation or visit the Microsoft Knowledge Base on line. This article may answer your questions:

http://support.microsoft.com/support/kb/articles/Q100/8/43.asp?LN=EN-US&SD=gn&FR=0&qry=Environment&rnk=11&src=DHCS_MSPSS_gn_SRCH&SPR=NTW40 Or search for: "Environment Variables in Windows NT."

For easy updating, the run script can sometimes be separated into two files. The first file passes variables to the next. Below is a typical file used which passes parameters (%1,%2, %3) to a second run script shown on Appendix I. The run script illustrated in Appendix I is receiving parameters from the file below.

```

:: Format of lines below:
:: Call [batchfilename] [release] [abroot] [server] [database] [userid] [encryption (True or false)]

call common50b_sybase.bat V52R2_08072001_BUILD_1047 test52 olf119 test52r2 trader1 TRUE

start "OLISTEN Console - Close Manually" olisten

```

Argument	Description
Cut	The path where the Endur/Findur directory is found
Abroot	The root directory where Endur/Findur resides
Server	The name of the user's server when they log in
Database	The name of the database you are linking to
UserId	The user's id:
Encryption	The required encryption parameter. Usually set to TRUE.

5.3.2 Licenses Required

- Toolkit developers and application users need an Endur/Findur and a Toolkit license.
- Toolkit developers need a Microsoft Visual Basic license.
- For additional requirements, such as Rendezvous, for running Endur/Findur, consult the Endur/Findur documentation.
- Licenses for other third-party software like MS Excel and Seagate Software Crystal Reports may also be required.

6. OPERATIONAL MODES

Visual Basic applications created with Toolkit can be run in two modes, Attached and Stand Alone. They are available as follows:

Version	Attached Mode	Stand-alone Mode
Enterprise TK Standard	Yes	No
Enterprise TK Pro	Yes	Yes

6.1 The Attach Mode of Operation

6.1.1 What Is Attach Mode?

In the Attach mode, you run an Endur/Findur session simultaneously with the Visual Basic application and the user is able to utilize both the Endur/Findur screens and the new application. Data entered in the Visual Basic application can be seen in the Endur/Findur screens.

6.1.2 Why Run in Attach Mode?

Advantages

- Add or modify data using the Visual Basic application and view the results on Endur/Findur's screens.
- View tables created in their Toolkit application by calling the TABLE_ViewTable function.
- Work with existing Endur/Findur screens in addition to the custom developed Toolkit Applications.
- Take advantage of Visual Basic's advanced debugging.

Disadvantages

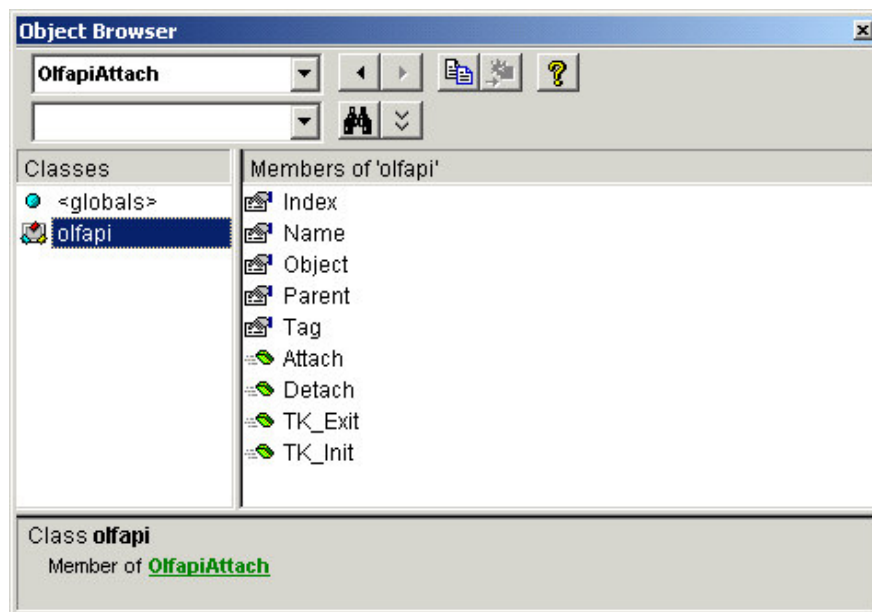
- Endur/Findur trading and risk management engines cannot be embedded into the Visual Basic custom application running in Attached mode, since the Endur/Findur screens are always exposed to the user. So you cannot create applications that run alone. Endur/Findur must be running as a task along side your application.
- The Visual Basic application that is designed to run in Attach mode is dependent on the Endur/Findur session. If the Endur/Findur application is terminated you will be unable to call Toolkit functions.

6.1.3 The OlfapiAttach.ocx Object

The OlfapiAttach.ocx file is used to attach to the locally running Endur/Findur session for development and execution of any custom developed Toolkit application in olfapi_reporting directory shipped with the Enterprise Toolkit application. We recommend that the task of registering and unregistering the OlfapiAttach.ocx at your location be carried out by the system administrator according to the Installation Guide. However, in this class we will practice the procedure of registering and unregistering the OlfapiAttach.ocx to illustrate the importance of this ActiveX object to Toolkit.

The OlfapiAttach.ocx object for Toolkit compatible with Endur/Findur 5.0 has the following properties, methods and events that can be browsed using the Object Browser.

⚡ The figure below is not intended as an exercise. It is for illustrative purposes only. Before you can view the object's properties using Visual Basic as shown below you will need to have followed the steps in section 6.1.5 to register the .ocx object and add it to the toolbar. Only when the object is visible in the toolbar can you view its properties.



Here is the list of olfapiAttach functions or methods included in the OlfapiAttach.ocx object:

Attach([program_name As String], [control_flag As Long], [change_dir As string]) As Long

This function attaches to a running session of Endur/Findur.

Detach() As Long

This function detaches from a running session of Endur/Findur.

TK_Exit() As Long

This function exits or closes the stand-alone session.

TK_Init(user As String, password As String, database As String, server As String, [change_dir As String])
As Long

This session starts the stand-alone mode.

6.1.4 The Attach Method of the OlfapiAttach.ocx

The Attach method takes 3 optional arguments, defined below:

```
olfapi1.Attach( [program_name As String], [control_flag As Long], [change_dir As String])
```

program_name (string)	A string reference to your Toolkit session
control flag (integer)	Should be set to either 0 or 1. The default value is 0. If control_flag = 1, your application will be a child process of master.exe (Endur/Findur engine) so your program will end when the locally running Endur/Findur session ends. If control_flag = 0, it runs independent of master.exe, so your program will continue even after the Endur/Findur session ends.
Change_dir (string)	String name for the path of your Endur/Findur bin directory. If change_dir is not specified, your program will assume your current directory is the Endur/Findur bin directory. This parameter is needed if the application is started from a prompt that is not in the machine's /bin directory. This parameter is used to change directory to the directory containing the Toolkit dll files needed to run the application. This becomes useful, when developing one application that can run across several Endur/Findur releases. This parameter is not a substitution for a properly configured environment

6.1.5 Developing and Running Attach Mode Applications

The following is a generalized procedure for running applications in the Attach mode. A complete example follows in section 6.1.6. Please note that before any of these procedures can be run successfully, the system administrator must first configure the workstation properly for Toolkit. Configuration is covered in the Toolkit installation guide.

Overview of Recommended Procedures

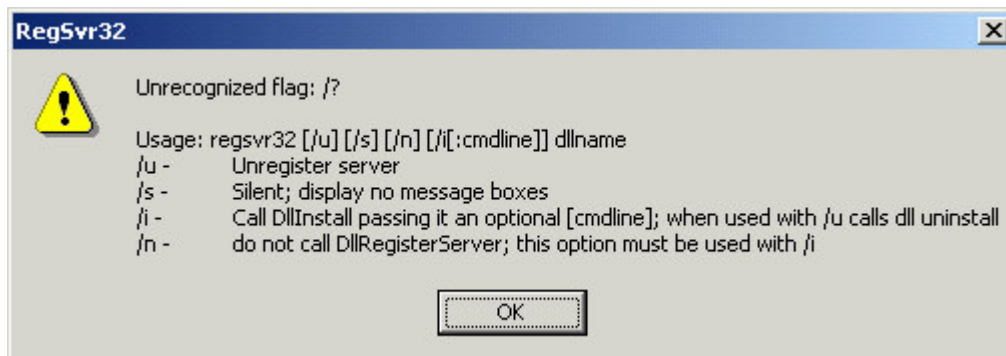
The Toolkit installation guide recommends that the OlfapiAttach.ocx is first copied to the local machine (e.g.\Winnt\System32) and registered from that location. However, you should always contact your systems administrator before copying any programs to your drive. Your local procedures may differ.³

It is important to record where your registered OlfapiAttach.ocx resides in your hard drive and if it was registered. We will cover this later.

If your machine had prior installations of toolkit, the system administrator must first unregister any existing OlfapiAttach.ocx entries. This process will remove the references from the Windows registry. Next, the administrator should delete the OlfapiAttach.ocx and the OlfapiAttach.oa files, if they exist. This should be done before copying the new OlfapiAttach.ocx to the local computer.

If this is not done, the next time you register an OlfapiAttach.ocx, Windows will create another registry entry for this OlfapiAttach.ocx and when you run Toolkit you may get an error. The system administrator will also have to make sure that all registry entries referring to the OlfapiAttach.ocx are removed, including those pointing to non-existing directories, known as "dead registry entries". This may require registry cleanup, a procedure that should be carried out by system administrator using an appropriate registry cleanup tool and is necessary only when serious problems are encountered.

Registration of the OlfapiAttach.ocx file is accomplished via the regsvr32 command issued from the command prompt window. The command must be entered at the directory location or path where the OlfapiAttach.ocx file exists. In this case, the local folder where you copied the OlfapiAttach.ocx to during the installation process. The same command is used with the /u option (unregister option) to unregister the file. Details of the command are obtained by typing regsvr32 /? at the command prompt to display the following:



³ Review the Installation Guide included on your CD under the topic OlfapiAttach.ocx

Note that by adding the /u option, you can unregister the application.

Suggested Procedure for Registering OlfapiAttach.ocx

⚡ This procedure is generic to registering any ActiveX control in Visual Basic. Please refer to a Visual Basic reference for additional information. In addition, this procedure should normally be carried out by the system administrator according to the recommendations in the Installation Guide, but we are reviewing this procedure here for illustrative purposes.

To register the file, follow these steps:

- 1.** Using the Windows explorer, navigate to the current release of Endur/Findur that contains the Toolkit software and open the olfapi_reporting folder. Copy the OlfapiAttach.ocx file to a folder on your local machine and note down the path of the folder where it was copied (e.g. c:/Winnt/system32).
- 2.** Click the **Start** button and select **Run**.
- 3.** Type **cmd** to display the command prompt and navigate to the location on your hard drive where the OlfapiAttach.ocx file in your local machine resides.
- 4.** Type **cd** (change directory) followed by the directory path where the latest version or release of Toolkit resides in and change to the directory in your local drive where the copy of the Olfapi_attach.ocx resides (e.g. c:/Winnt/system32).
- 5.** Then type **regsvr32 OlfapiAttach.ocx** from a command line. A message box will appear to inform you that the registration was successful.

⚡ Please be sure you have unregistered all prior OlfapiAttach.ocx files before you register a new one.

Example:


From the directory where OlfapiAttach.ocx exists, type the following at the prompt to register the attach ActiveX control.

```
Regsvr32 OlfapiAttach.ocx
```

Running in Attach Mode

The following is a generalized procedure for running any application in Attach mode. The procedure requires you to run Endur/Findur first from the command prompt, then run Visual Basic from the same prompt and then run Example1, which contains the Visual Basic code that Attaches to Endur/Findur.

1. First, you will code the Visual Basic application using the Attach code subroutine illustrated in Example 1, section 6.1.6 below. You can vary the code to include different forms and events to activate the attaching function. For this training course, we will run Example 1 after we follow the steps below.
2. Go to the **Start** button in Windows, select **Run** and type **cmd**. The command window will appear.

 **Important:** It is crucial that Endur/Findur and Visual Basic are run from the same command prompt or run script and not run as separate processes directly from Windows. This ensures that the system configures the proper environment (Endur/Findur and Visual Basic must have the same environment settings) and that the examples run correctly. Running Endur/Findur and Visual Basic as separate processes directly from Windows could result in errors.



Tip: On the command window, right click on the title bar along the top of the window and select Properties from the menu. Click the Layout tab from the menu. In the Buffer Size area, type 1000 into the Height box. Then, click OK at the prompt. Now you will be able to scroll up the window and view the history of commands.

3. Type **CD** and the path of the directory where the Endur/Findur run script used by your firm resides:


Example:

```
S:
CD S:/abacus/main/test_runscripts
```

The above example changes to the directory where the script is found.



Tip: Type in the command TREE to view the directory structure graphically within the command prompt window.

 *What's in the run script? See Appendix I.*

4. Run the **Endur/Findur** session by typing the name of the run script.

Example:

```
runtest50r4
```

☞ The above command runs the batch file (.bat) or script named runtest50r4.bas, which sets all of the environment variables and runs Endur/Findur.

5. Type in your username and password and run Endur/Findur as usual.

6. Go back to the **cmd** prompt and type **VB6** to run Visual Basic.

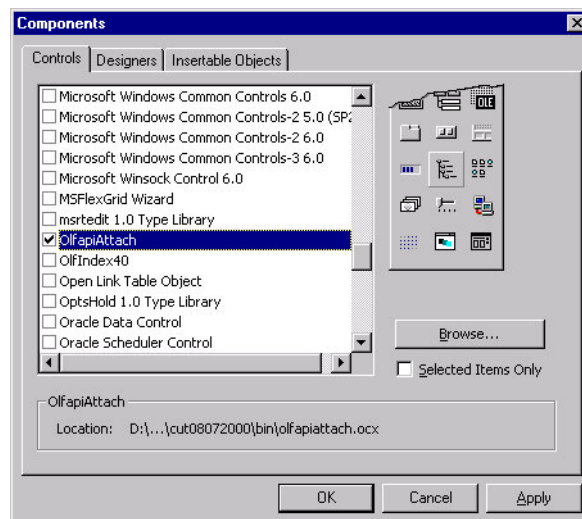
☞ Do not click on Visual Basic from Windows. If Visual Basic does not run, have your system administrator check to make sure the VB6.exe executable is your System Path.

7. When Visual Basic runs, load and run your example or application from Visual Basic.

☞ If you plan to compile the application, save it in your drive. Then, run it from the command prompt after you run the Endur/Findur runscript.

Adding the OlfapiAttach.ocx Control

A control can be added to the “Toolbox” for use in a project by right clicking on the “Toolbox” and selecting “Components” from the pop-up menu. Scroll the components list until the OlfapiAttach.ocx control is visible and click on it to add it to the toolbox.



Notice that OlfapiAttach.ocx is checked off. Click “OK.” You will also note that the object now appears in the toolbox.

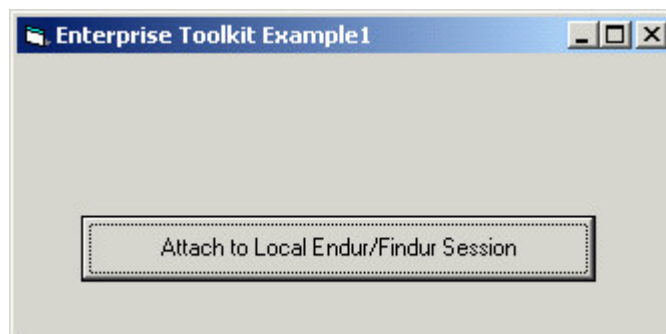
☞ Form1, in Example 1 illustrated below contains olfapi1, an instance of the OCX, and a command button. That object was dragged in from the toolbox. If you are working with a new version of the OlfapiAttach.ocx, you will have to remove that instance of the OlfapiAttach.ocx and drag it on the form to create a new instance or copy containing the new code changes. You will need to do this for any program where the OlfapiAttach.ocx appears on a form as shown whenever a new OlfapiAttach.ocx release is issued. To avoid this, consider using Create Object as explained in section 6.1.8 below.

6.1.6 Example 1: Attach to Endur/Findur

The following example illustrates the code required in order to attach to a running session of Endur/Findur.

*⚡ Make sure you have registered the latest version of **OlfapiAttach.ocx** as indicated above and then follow steps 1-7 above to run Endur/Findur and Visual Basic (from the CMD prompt.)*

1. From Visual Basic, select **Open Project** from the **File** menu and go the Examples directory.
2. Click the **Examples** folder and then click the **Example 1** folder. Click to highlight the file **Project 1.vbp**.
3. Click **Open**. The file will load to your Visual Basic project. In the project explorer, right click the form to reveal the code.
4. From the **Run** menu, select **Start** or press **F5** to run the code. You will see the following:



5. Click the button and wait for a response. A message "Attach Method Succeeded" will be returned after some time.

The code is shown on the next page followed by a diagram illustrating the basic components of this code snippet.

*⚡ **Important:** This prepackaged example already has the OlfapiAttach.ocx object dropped into the form*

6. After successfully running Example 1, edit the code and change the following line of code from:

```
retval = olfapi1.Attach("Enterprise TK Example1", 0, strDir)
to
retval = olfapi1.Attach("Enterprise TK Example1", 1, strDir)
```

7. Compile and save and run Example 1 and observe what happens to this VB application when you exit Endur/Findur.

6.1.7 Example 1 Code

```

'*****
'*
'*          OpenLink Financial, Inc.
'*
'*          Copyright (C) 1992 - 2001
'*          ALL RIGHTS RESERVED
'*
'*          Toolkit Example1
'*          This Example requires Endur/Findur with the Toolkit Already Setup.
'*
'* Objective:  * Successfully Attach the Toolkit to Endur/Findur.
'*            * Learn how to use the OlfapiAttach.ocx user control
'*
'*****
Option Explicit

-----
' Name: cmdAttach_Click
'
' Description: This sub attaches to the locally running Endur/Findur session
'              and initializes the example
' Developer:  Toolkit Developer
-----

Private Sub cmdAttach_Click()
    Dim retval As Long
    Dim strDir As String
    On Error GoTo HandleErrors
    'The two lines below should be uncommented if you wish to create the olfapiAttach.ocx object instead of using the
    'object on the form.
    ' Dim myobject As Object
    ' Set myobject = CreateObject("OlfapiAttach.olfapi")

    'The following code is used to get the Endur/Findur bin directory.
    'This directory must be entered for the OlfapiAttach.ocx to be able to attach.
    strDir = Environ("AB BIN")
    If strDir = "" Then
        strDir = InputBox("Please Enter Your Endur/Findur Bin Directory:", "OpenLink Financial", CurDir)
        If strDir = "" Then
            MsgBox "You Must Enter A Endur/Findur Bin Directory", vbInformation, "OpenLink Financial"
            Exit Sub
        End If
    End If

    retval = olfapi1.Attach("Enterprise TK Example1", 0, strDir) 'This is the basic attach command.

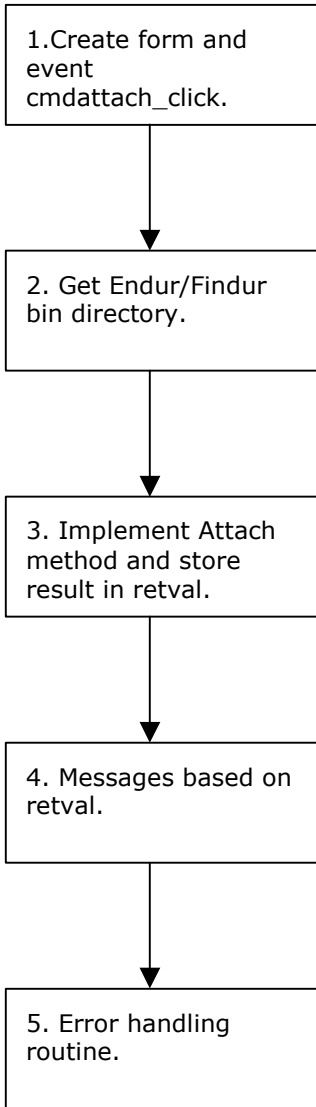
    ' retval = myobject.Attach("Enterprise TK Example1", 0, strDir)
    'The Attach method takes 3 optional arguments...
    'olfapi1.Attach( [program name As String], [control flag As Long], [change dir As String])
    ' * The program name is just a string reference to your Toolkit session.
    ' * The control flag should be set to either 0 or 1.
    '   0 - if your application will run independent of master.exe
    '   1 - if your application will be a child process of master.exe
    ' * The change dir argument is the path of your Endur/Findur bin directory. This is necessary
    ' for the Toolkit to attach.

    If (retval = 0) Then
        'Attach method has failed.
        MsgBox "Attach Method Has Failed", vbInformation, "Enterprise TK Example1"
    ElseIf (retval = 1) Then
        'Attach method has succeeded.
        MsgBox "Attach Method Has Succeeded", vbInformation, "Enterprise TK Example1"
    ElseIf (retval = 2) Then
        'Application is already attached.
        MsgBox "Already Attached", vbInformation, "Enterprise TK Example1"
    End If
    Exit Sub

HandleErrors:
    'A common error that may be experienced is ...
    ' olfapi_reporting.dll not found.
    'If you get this error, it means that either your path is not set correctly, a dll that
    'the Toolkit is dependent on is missing or possibly that the Toolkit dll version is not
    'correct for the version of Endur/Findur. A helpful tool in dealing with this problem is the
    'Microsoft Depends application that is part of MS Visual Studio 6.0 Tools. It should
    'let you know what dll may be missing or what dll version may not be correct.
    MsgBox "Error Number: " & Err.Number & vbCrLf &
        Err.Description, vbCritical, "Enterprise TK Example1"
End Sub

```

Example 1 Attaching to Running Session of Endur/Findur



```

'*****
'*
'*          Open Link Financial, Inc.
'*
'*          Copyright (C) 1992 - 2001
'*          ALL RIGHTS RESERVED
'*
'*          Toolkit Example
'*          This Example requires Endur/Findur with the Toolkit Already Setup.
'*
'* Objective:  * Successfully Attach the Toolkit to Endur/Findur.
'*            * Learn how to use the OlfapiAttach.ocx user control
'*
'*****
Option Explicit

'-----
' Name: cmdAttach_Click
' Description: This sub attaches to the locally running Endur/Findur session
'              and initializes the example
' Developer:  Toolkit Developer
'-----

Private Sub cmdAttach_Click()
    Dim retval As Long
    Dim strDir As String
    On Error GoTo HandleErrors
    'The two lines below should be uncommented if you wish to create the olfapiAttach.ocx object in
    'object on the form.
    \ Dim myobject As Object
    \ Set myobject = CreateObject("OlfapiAttach.olfapi")

    'The following code is used to get the Endur/Findur bin directory.
    'This directory must be entered for the OlfapiAttach.ocx to be able to attach.
    strDir = Environ("AB BIN")
    If strDir = "" Then
        strDir = InputBox("Please Enter Your Endur/Findur Bin Directory:", "Open Link Financial")
        If strDir = "" Then
            MsgBox "You Must Enter A Endur/Findur Bin Directory", vbInformation, "Open Link Fin
            Exit Sub
        End If
    End If

    retval = olfapi1.Attach("Enterprise TK Example1", 0, strDir) 'This is the basic attach com
    \ retval = myobject.Attach("Enterprise TK Example1", 0, strDir)

    'The Attach method takes 3 optional arguments...
    'olfapi1.Attach( [program name As String], [control flag As Long], [change dir As String])
    ' * The program name is just a string reference to your Toolkit session.
    ' * The control flag should be set to either 0 or 1.
    '   0 - if your application will run independent of master.exe
    '   1 - if your application will be a child process of master.exe
    ' * The change dir argument is the path of your Endur/Findur bin directory. This is nec
    ' for the Toolkit to attach.

    If (retval = 0) Then
        'Attach method has failed.
        MsgBox "Attach Method Has Failed", vbInformation, "Enterprise TK Example1"
    ElseIf (retval = 1) Then
        'Attach method has succeeded.
        MsgBox "Attach Method Has Succeeded", vbInformation, "Enterprise TK Example1"
    ElseIf (retval = 2) Then
        'Application is already attached.
        MsgBox "Already Attached", vbInformation, "Enterprise TK Example1"
    End If
    Exit Sub

HandleErrors:
    'A common error that may be experienced is ...
    ' olfapi_reporting.dll not found.
    'If you get this error, it means that either your path is not set correctly, a dll that
    'the Toolkit is dependent on is missing or possibly that the Toolkit dll version is not
    'correct for the version of Endur/Findur. A helpful tool in dealing with this problem is t
    'Microsoft Depends application that is part of MS Visual Studio 6.0 Tools. It should

```

6.1.8 Creating the OlfapiAttach.ocx Versus Dropping the Object

The OlfapiAttach.ocx can be implemented in two ways in a Visual Basic application:

1. **Drop The Object method:** where you add the OlfapiAttach.ocx to the toolbar, as indicated on page 35, and then drop it on the form.
2. **Create Object method:** where the object is created from within the application. You do not need to drop the object to the form if you use the following code:

```
Dim myobject As Object
Set myobject = CreateObject("OlfapiAttach.olfapi")
```

For simplicity in running the examples in this course, most of the examples, with the exception of the reporting example in Module IV, use the Drop The Object method. However, in many complex coding applications OpenLink recommends using the Create Object function instead. It is up to the developer to determine what method to use. The table below details the advantages and disadvantages of both methods.

Method	Advantages	Disadvantages
Drop The Object	<p>Best for coding applications, which use events recognized by the olfapiattach.ocx object.</p> <p>Note: The olfapiattach.ocx shipped with Toolkit Solutions for Endur/Findur versions v5.2 (and up) recognizes Endur/Findur system events.</p> <ul style="list-style-type: none"> - Master Down - Error Event 	<p>Creates a "hard reference" to a specific olfapiattach.ocx object. This may require dropping a new olfapiattach.ocx onto existing VB application forms if future registrations of a new olfapiattach.ocx are necessary.</p>
Create Object	<p>The "soft reference" to the olfapiattach.ocx object allows VB applications to work without revision if future registrations of a new olfapiattach.ocx are necessary.</p>	<p>Applications cannot be coded to utilize Endur/Findur system events recognized by the olfapiattach.ocx object as shipped with Toolkit Solutions package for Endur/Findur v5.2 (and up).</p>

6.2 Stand-alone Mode

6.2.1 What is Stand-alone Mode?

This mode of operation may be necessary to create custom applications that completely embed the Endur/Findur trading and risk management engines in their application. Applications run as stand alone, without the need of running Endur/Findur in the machine and viewing the user screens. The most notable difference when running in this mode is that the familiar Endur/Findur Master Interface is ABSENT.

🔗 Note: Stand-alone operation is available only with Enterprise TK Pro™.

6.2.2 Why Run in Stand-alone Mode?

This mode of operation is essential when using Toolkit to implement interfaces to external systems that can operate in unattended mode.

Advantage

Enables true embedding of Endur/Findur suite of trading and risk management features by developing engines that allow development of customized and proprietary graphical interfaces.

Disadvantage

Not ideally suited for debugging. For example, to view a table from your application, you will need to write your own routine for displaying the data in the table.



Alternatively, if you are the developer and have the source code to the application, simply switch to Attach mode when you want to debug.

6.2.3 The Stand-alone Method

There are two ways to run a stand-alone application:

- The TK_Init function of the OlfapiAttach.ocx control.
- The OLF_EnterpriseTKInit function from the function library.

TK_Init function of the OlfapiAttach.ocx

The preferred method is to use the TK_Init function of the olfapiAttach.ocx described in section 6.1.3 The OlfapiAttach.ocx Object. In Visual Basic this is implemented as an instance of the object as follows:

```
retVal = olfapi1.TK_Init(username, password, database, server, _ab_root_string)
```

These are the return values of this method:

-1	DLL is already initialized. Or This may mean that Master is running.remove file <lock_filename>;
-2	TEMP environment not set.
-3	Cannot create shared memory
-4	Cannot get or use environment variables
-5	SSQL error returned trying to log in to DB.
-7	Password Expired - Please change your password.
-8	Your unmarks (permission bits mask) is set incorrectly. Please consult your system administrator.
-9	Error in user environment.
-10	Error occurred setting Index Tables.
-11	Error occurred initializing currencies.
-12	Error occurred initializing volatilities.
-14	Unable to init message center
-15	failed Task Manager Init
-16	Unable to start master beacon

The OLF_EnterpriseTKInit function

The other method, used in older installations is the **OLF_EnterpriseTKInit** function where the .ocx may not support the above method.

In order to call the method, the **olfapi.bas** module file, which contains the prototype for this method, *must* be included in an application running in the Stand-alone mode and the **OLF_EnterpriseTKInit** method *must* be called before any other function call is made or an error will occur. **OLF_EnterpriseTKExit** is used to detach from the Enterprise engines.

The **OLF_EnterpriseTKInit** method takes seven parameters: user, password, server, database, Endur/Findur directory, error message, and maximum error length.

```
OLF_EnterpriseTKInit( [user], [password], [database], [server], [cut dir], [err_msg],
[max_err_len])
```

Argument/Parameter	Description
user	User name string variable
password	User password string variable
server	Server string variable where the database is located
database	Database name
cut dir	Directory path variable describing the location of the Endur/Findur User, Sys, and Out directories
Error_msg	String variable to hold an error message returned by an initialization failure
max_err_len	Integer variable describing the maximum length of an error message

The OLF_EnterpriseTKInit method returns an integer code to indicate the success or failure of Endur/Findur session initialization attempt.

Return Value	Description
< 1	Indicates the initialization was not successful
-1	DLL is already initialized. Or remove file <lock_filename>
-2	TEMP environment not set.
-3	Cannot create shared memory
-4	Cannot get or use environment variables
-5	SSQL error returned trying to log in to DB
-7	Password Expired - Please change your password.
-8	Your umask (permission bits mask) is set incorrectly. Please consult your system administrator.
-9	Error in user environment
-10	Error occurred setting Index Tables.
-11	Error occurred initializing currencies
-12	Error occurred initializing volatilities
-14	Unable to init message center
-15	Failed Task Manager Init.
-16	Unable to start master beacon.
1	Indicates the initialization was successful.

6.2.4 Example of TK_Init Code

The following example illustrates the code required in order to run in Stand-alone mode. The example can be used as a template for your own applications. Just cut and paste. Note that the examples shows two ways of implanting the function. You can call the OLF_EnterpriseTKInit function directly or you can drop the olfapiAttach object to the form and implement the TKInit property of it. Practice both forms.

1. Close any currently running Endur/Findur session on your PC/Workstation.
2. Close any currently running VB6 session on your PC/Workstation.
3. Make sure you have registered the latest version of **OlfapiAttach.ocx** as indicated in section 6.1.5.
4. Create a directory in your hard drive called tmp.

5. Copy both runscripts (e.g. common50_sybase.bat and runtest50r4.bat) to the local tmp directory.
6. Right click on the batch initialization file (the long .bat file, e.g. common50b_sybase.bat) and select **edit**.
7. Comment out the line that runs **Master.exe** and add a line to run VB.
For example,

```
:: start "Enterprise %1 CONSOLE " master -u %5  
start vb6
```

☞ Adding the Visual Basic line to the script ensures that the correct environment variables are set prior to running your applications.

8. Save the file and close. Note the name of the file.
9. If you are working with a two-part runscript, right click on the run script file in the **tmp** directory you created and select **Edit**. Look at the file name after the Call statement. Make sure it is the same name as the file you saved in the prior step.

E.G. Call common50b_sybase.bat

10. Save and close the file.
11. Click on the runscript file name in your tmp directory to start Endur/Findur.
12. From Visual Basic, select New Project and then click Standard.exe and click the Open Button. Select the prepackaged Example1a provided by your instructor.
13. As you load the example, click OK to messages that refer to the .ocx not found. You will be adding this control to your project and to the form. See the next section.

```

*****
'*
'*                               Open Link Financial, Inc.
'*
'*                               Copyright (C) 1992 - 2001
'*                               ALL RIGHTS RESERVED
'*
'*                               Enterprise Toolkit Example1a
'*                               This Example requires Endur/Findur with the Enterprise Toolkit Already Setup.
'*
'* Objective:      * Successfully Initialize the Enterprise Toolkit Pro
'*                  * for stand-alone mode to Endur/Findur.
'*                  * Learn how to use the OlfapiAttach.ocx user control.
'*
*****
Option Explicit
'Application Name For Message Boxes.
Const APP_NAME As String = "Enterprise Toolkit Example1A"

'-----
' Name: Form_Load()
'
' Description: This sub initializes the Enterprise Toolkit Example1A Form.
' Developer:  Toolkit Developer
'-----
Private Sub Form_Load()
    'The following statements initialize values in the text boxes based off of Environment variables if
    they are set.
    txtBin.Text = Environ("AB_BIN")
    txtServer.Text = Environ("AB_LOGON_SERVER")
    txtDatabase.Text = Environ("AB_LOGON_DATABASE")
End Sub

'-----
' Name: cmdTKInit_Click()
'
' Description: This sub initializes the Enterprise TK Pro for stand alone mode.
' Developer:  Toolkit Developer
'-----
Private Sub cmdTKInit_Click()
    Dim lngRetVal As Long

    On Error GoTo HandleErrors

    'The following constants represent the possible return values from the TK_Init Call.
    Const OLF_FAIL = 0
    Const OLF_SUCCEED = 1
    Const OLF_ALREADY_INITED = 2
    Const OLF_FAIL_MASTER_RUNNING = -1
    Const OLF_FAIL_TEMP_ENV_NOT_SET = -2
    Const OLF_FAIL_CAN_NOT_CREATE_SHARED_MEM = -3
    Const OLF_FAIL_GET_USE_ENV_VARS = -4
    Const OLF_FAIL_SSQL_ERROR = -5
    Const OLF_FAIL_PASSWORD_EXPIRED = -7
    Const OLF_FAIL_UMASK_ERROR = -8
    Const OLF_FAIL_ERROR_IN_ENV = -9
    Const OLF_FAIL_SETTING_INDEX_TABLES = -10
    Const OLF_FAIL_INITTING_CURRENCIES = -11
    Const OLF_FAIL_INITTING_VOLS = -12
    Const OLF_FAIL_INITTING_MSG_CENTER = -14
    Const OLF_FAIL_INITTING_TASK_MGR = -15
    Const OLF_FAIL_INABLE_TO_START_MASTER_BEACON = -16

    'DLL is already initialized. Or remove file
    '<lock filename>.
    'TEMP environment not set.
    'Cannot create shared memory.
    'Cannot get or use environment variables.
    'SSQL error returned trying to log in to DB.
    'Password Expired - Please change your password.
    'Your umask (permission bits mask) is set
    'Error in user environment.
    'Error occurred setting Index Tables.
    'Error occurred initializing currencies.
    'Error occurred initializing volatilities.
    'Unable to init message center.
    'Failed Task Manager Init.
    'Unable to start master beacon.

```

```

'This is where the Enterprise TK Pro actually initializes itself. It requires the user to pass in a
'username, password, database, server and an optional bin directory.
lngRetVal = olfapil.TK_Init(txtUsername.Text, txtPassword.Text, txtDatabase.Text, txtServer.Text,
txtBin.Text)
Select Case lngRetVal
  Case OLF_FAIL:
    'The toolkit has failed to attach. Possible reasons for this result are that certain binary
files (dll's)
    'are missing, the user's path is not correct or the Enterprise Toolkit Pro is not properly
licensed.
    MsgBox "Failed to Initialize TK Stand alone", vbCritical, APP_NAME
    End

  Case OLF_SUCCEED:
    'The Enterprise Toolkit Pro has successfully initialized for stand alone mode.
    MsgBox "Successfully initialized Enterprise TK Stand Alone Mode.", vbInformation, APP_NAME

  Case OLF_ALREADY_INITED:
    'The Enterprise Toolkit Pro has already been initialized for stand alone mode.
    MsgBox "Application has already initialized Enterprise TK Stand Alone Mode.", vbInformation,
APP_NAME

  Case OLF_FAIL_MASTER_RUNNING:
    'DLL is already initialized. Or remove file <lock_filename>
    MsgBox "DLL is already initialized. Or remove file <lock_filename>.", vbInformation, APP_NAME
    End

  Case OLF_FAIL_TEMP_ENV_NOT_SET:
    'TEMP environment not set.
    MsgBox "TEMP environment not set.", vbInformation, APP_NAME
    End

  Case OLF_FAIL_CAN_NOT_CREATE_SHARED_MEM:
    'Cannot create shared memory
    MsgBox "Cannot create shared memory.", vbInformation, APP_NAME
    End

  Case OLF_FAIL_GET_USE_ENV_VARS:
    'Cannot get or use environment variables
    MsgBox "Cannot get or use environment variables.", vbInformation, APP_NAME
    End

  Case OLF_FAIL_SSQL_ERROR:
    'SSQL error returned trying to log in to DB.
    MsgBox "SSQL error returned trying to log in to DB.", vbInformation, APP_NAME
    End

  Case OLF_FAIL_PASSWORD_EXPIRED:
    'Password Expired - Please change your password.
    MsgBox "Password Expired - Please change your password.", vbInformation, APP_NAME
    End

  Case OLF_FAIL_UMASK_ERROR:
    'Your umask (permission bits mask) is set incorrectly. Please consult your system
administrator.
    MsgBox "Your umask (permission bits mask) is set incorrectly. Please consult your system
administrator.", vbInformation, APP_NAME
    End

  Case OLF_FAIL_ERROR_IN_ENV:
    'Error in user environment.
    MsgBox "Error in user environment.", vbInformation, APP_NAME

```

```

End

Case OLF_FAIL_SETTING_INDEX_TABLES:
'Error occurred setting Index Tables.
MsgBox "Error occurred setting Index Tables.", vbInformation, APP_NAME
End

Case OLF_FAIL_INITING_CURRENCIES:
'Error occurred initializing currencies.
MsgBox "Error occurred initializing currencies.", vbInformation, APP_NAME
End

Case OLF_FAIL_INITING_VOLS:
'Error occurred initializing volatilities.
MsgBox "Error occurred initializing volatilities.", vbInformation, APP_NAME
End

Case OLF_FAIL_INITING_MSG_CENTER:
'Unable to init message center
MsgBox "Unable to initialize message center.", vbInformation, APP_NAME
End

Case OLF_FAIL_INITING_TASK_MGR:
'Failed Task Manager Init
MsgBox "Failed to initialize Task Manager.", vbInformation, APP_NAME
End

Case OLF_FAIL_INABLE_TO_START_MASTER_BEACON:
'Unable to start master beacon
MsgBox "Unable to start master beacon.", vbInformation, APP_NAME
End

Case Default:
'An unsupported return value has been received.
MsgBox "Return Value " & lngRetVal & " has been returned. TK_Init Has Failed.",
vbExclamation, APP_NAME
End
End Select
Exit Sub
HandleErrors:
MsgBox "An Error Has Occurred While Trying To Perform TK_Init." & vbCrLf & _
"Error Number: " & Err.Number & vbCrLf & Err.Description, vbInformation, APP_NAME
End
End Sub

'-----
' Name: cmdTKExit_Click()
'
' Description: This sub Exits the Enterprise TK Pro from stand alone mode.
' Developer: Toolkit Developer
'-----
Private Sub cmdTKExit_Click()
'It is very important to call TK_Exit every time you do a TK_Init to free shared memory.
olfapil.TK_Exit
End Sub

'-----
' Name: Form_Unload(Cancel As Integer)
'
' Description: This sub Exits the Enterprise TK Pro from stand alone mode.
' Developer: Toolkit Developer
'-----
Private Sub Form_Unload(Cancel As Integer)

```

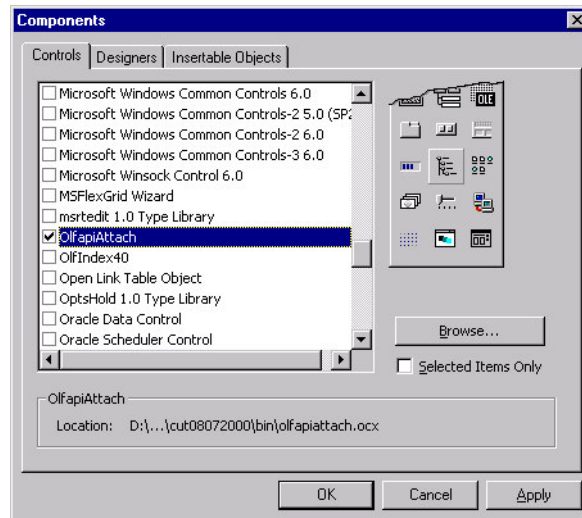
```

'Ve call this on the form unload just in the TK Exit Command button was not clicked. No harm is
'done when calling this more than once. It is very important to call TKExit every time you do a
TKInit
'to free shared memory.
olfapi1.TK_Exit
End Sub

```

1. Add the OlfapiAttach.ocx Control

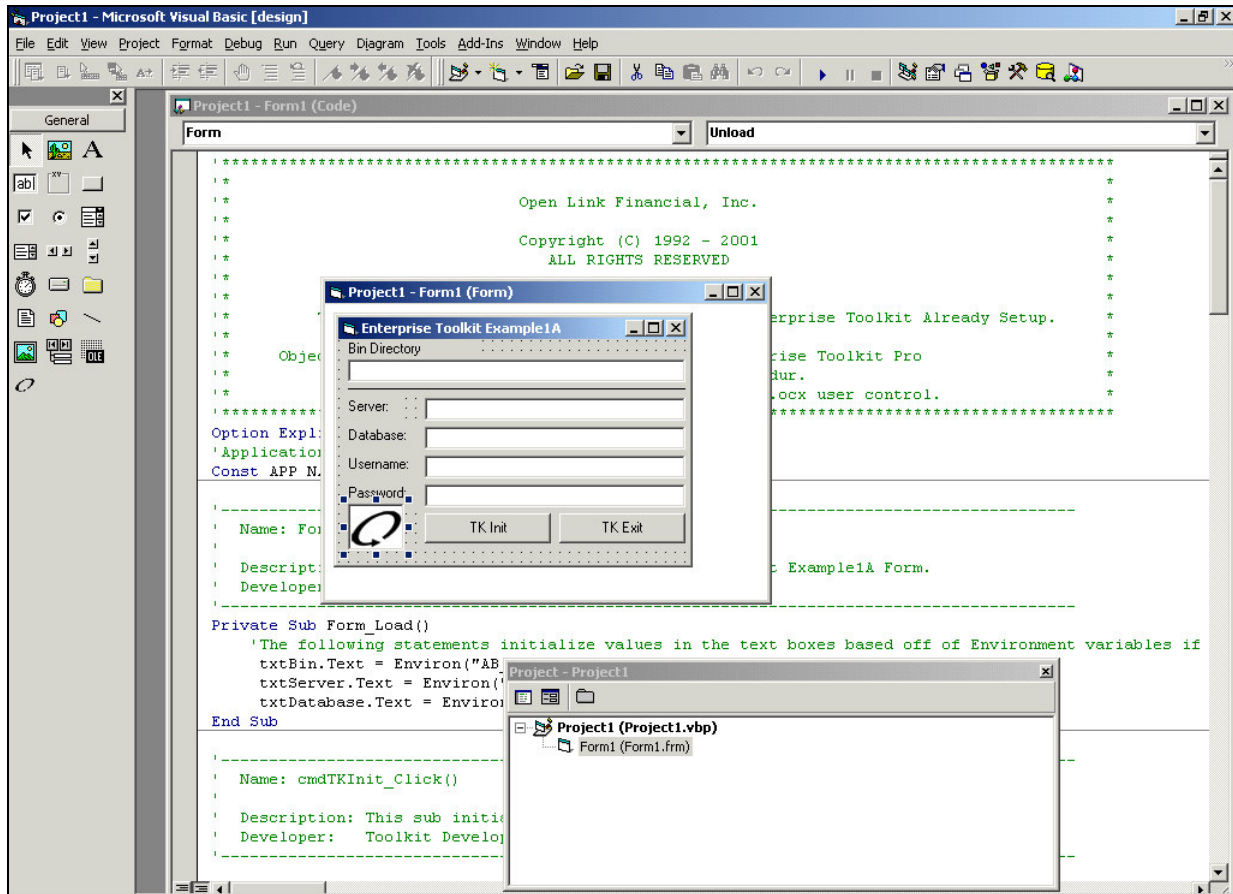
In order for Example1a to work, the OlfapiAttach.ocx control must be added to the "Toolbox" for use in a project by right clicking on the "Toolbox" and selecting "Components" from the pop-up menu. Scroll the components list until the OlfapiAttach.ocx control is visible and click on it to add it to the toolbox.



Notice that OlfapiAttach.ocx is checked off. Click "OK." You will also note that the object now appears in the toolbox.

2. Drag the OlfapiAttach.ocx to the form

In order for the TK_init method to work correctly, an instance of the OlfapiAttach.ocx needs to be created on the form. This is done by dragging and dropping the .ocx object from the toolbar to the form as shown below.



3. Now run the project. Fill in the path of the bin directory on your server, the server name, database name and your user name and password.
4. Press the TK_Init button.
5. To close the connection to the database press the TK_Exit button.



Setting the AB_SERVER environment variable to TRUE will suppress all warning and error messages and allow your Toolkit application to run unattended.



As an alternative, you may make an executable of the project, save it on your hard drive, and run it from the CMD prompt.

7. EXERCISES

Find the functions that operate on the Environment variables using the Toolkit Glossary navigator (the left pane of the Glossary window).

APPENDIX I – TYPICAL RUNSCRIPT FILE

Below is a typical runscript used to start Endur/Findur. Notice that all of the environment variables are set here. Comments in this file describe the variable's function. You can obtain a complete list of environment variables and definitions by clicking the Endur/Findur help file and searching the topic "Environment Variables."

You can see a list of values assigned to the environment variables by clicking the "i" button on the Endur/Findur system.

```

set AB_COMPANY=OLF
::
:: Directory where Abacus modules are installed
::
:: The binaries should be on a local area network fileserver.  If going across
:: a WAN, install on a local server or locally on the workstation and modify
:: those users run script accordingly.  This directory needs read/execute permission.
::
set AB_BIN=u:\builds\%1\bin

::
:: Abacus system, user and report directories
:: THESE DIRECTORIES MUST BE WRITABLE BY ALL USERS
::
:: These directories should be accessible by all users using the same database
:: share across a WAN also.
::
set AB_ROOT=u:\abacus\%2
set AB_OUTDIR=%AB_ROOT%\outdir
set AB_SYSDIR=%AB_ROOT%\sysdir
set AB_USERDIR=%AB_ROOT%\userdir
set AB_AIF_DIR=%AB_OUTDIR%

::
:: Printing environment: local to your print system
::
:: leave as is (commented out) unless you have some special print needs
::
::set AB_PRINTER="print /D:%s"

::
:: Third Party Software, Sybase, Rendezvous, Notes, etc. install dirs
::
:: Sybase used primarily for Open Client DLL's and SQL.INI file that describes the
server
:: RV=Tibco's Rendezvous software which is configured/licensed per subnet & O/S
:: NOTES (optional) directory is only required for user running notes_mgr (see
AB NOTES_USER below)
:: Crystal (optional) reports if 3rd party reporting is desired
:: Reuters (optional) for live market feeds - only required for mdd mgr (see
AB_MDD_REMOTE_DIST below)
::

```

```
set SYBASE=s:\sybase11
set RV=s:\rv6.5;s:\rv6.5\bin
set NOTES=c:\notes
set CRYSTAL_GLOBAL=s:\crystal7
set CRYSTAL_LOCAL=c:\Program Files\Seagate Software\Crystal Reports
set CRYSTAL=%CRYSTAL_GLOBAL%;%CRYSTAL_LOCAL%
set AB_CRYSTAL_DIR=%AB_ROOT%\crystal
set REUTERS=s:\reuters;s:\reuters4
set MSDEV=c:\program files\microsoft visual studio\vb98

::
:: The all important Path
:: SystemRoot is for finding print.exe
::
set
PATH=%MSDEV%;%AB_BIN%\sybase;%AB_BIN%;%RV%;%NOTES%;%SYBASE%\dll;%REUTERS%;%SystemRoot%\s
ystem32;%CRYSTAL%;c:\microsoft visual studio\vb98;c:\program files\microsoft visual
studio\vb98

::
:: Sybase server and database info
::
set AB_LOGON_SERVER=%3
set AB_LOGON_DATABASE=%4

::
:: Notes Related
::
set NOTES_SERVER_NAME=lnserver
set AB_NOTES_RICHTEXT_PS=8
set AB_NOTES_USER=%USERNAME%.%COMPUTERNAME%
::
:: Nueron Data specific
::
set ND_PATH=%AB_BIN%
set OIT_WINMGR=MSWINDOWS

::
:: Open interface settings for OI4.0
:: Allow for 256 color use on
::
set ND_PC256PALETTE=FORCEALL
set OIT_SCALEFACTORS=;90

::
:: Abacus Daemon Users
:: format is o/s username.hostname (CASE SENSITIVE!)
::
set AB_MDD_REMOTE_DIST=username.machinename

::
:: Importing Data in Ref mgr
::
::set AB_IMPORT_SEPARATOR=","
```

```
::set AB_IMPORT_STRIP_LEAD_TRAIL_SEPARATOR=no
::set AB_IMPORT_QUOTED_STRINGS=yes

::
:: Importing data in Market Manager
::
set AB_IMPORT_CLOSE=%AB_ROOT%\sysdir\import
set AB_IMPORT_INTRA=%AB_ROOT%\sysdir\import
set AB_IMPORTDIR=%AB_ROOT%\sysdir\import
::
:: Market Data Variables
::
set AB_MDD_DEFAULT_SOURCE=REUTERS
set AB_MDD_SCOPE=local
::set AB_MDD_SCOPE=remote
set AB_MKTD_DEFAULT_OFF=TRUE
set AB_TRIARCH_APP_NUM=35
set IPCROUTE=%REUTERS%\var\triarch\ipcroute
::
:: MAKE SURE TRIARCH_DEMO IS NOT SET UNLESS FOR A DEMO!
::
::set TRIARCH_DEMO=s:\reuters\demotickfile
::
:: TRIARCH_DEMO is to provide a sample Reuters file for demo purposes.

::
:: Misc Variables
::
set AB_SUPPRESS_VIEW_PRINT_OK=TRUE
set AB_SUPPRESS_PRINT_TICKET=FALSE
set AB_PARSER=DPARSER
set AB_LOAD_DEFAULT_DOCUMENTATION=1
set AB_SHMEM_SIZE=4000000
set BEACON_INTERVAL=30
set MAX_TASK_ENGINES=2
set AB_SUPPRESS_VIEW_PRINT_OK=TRUE
set AB_HELP_DIR=u:\abacus\helpdir50
::set AB_RV_NETWORK_FILE=s:\networks\netfile.dat
set AB_ALLOW_MULTI_SESSION=FALSE
set DISABLE_DEAL_ENTRY_ON_GAS_BLOTTER=TRUE
set AB_MQ_INBOUND_ERROR=better_ask_grossi
set AB_OLF_DEVELOPER=TRUE
set AB_AIF_DIR=u:\abacus\morgan_pm_test
::set AB_RV_NETWORK_FILE=S:\abacus\main\rv_network.txt

set AB_ENCRYPTION_FLAG=TRUE

echo "This is a %1 cut"

cd /D %AB_BIN%
start "Enterprise %1 CONSOLE " master -u %5
start vb6
```

⁴ If you modify the runscript file, you will have to restart the Endur/Findur and Visual Basic programs. The environment variable will be known to Visual Basic as long as you start it from the command prompt after you run the runscript.

⁵ You may set this environment variable on the Windows system by going to Start: Settings: Control Panel and click on System icon. Select the Advanced tab and click on Environment variables. You may add a new variable to this list by clicking the New button. You may need to reboot to set the variable.