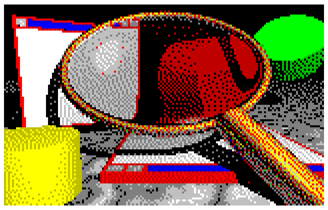
Project Analyzer 3.1



Overview of Project Analyzer

Menus

List procedures...
List variables and constants...
-
Name shadowing
Needless globals report
DLL report
-
Problem report
Design quality report
Project summary

Options

Show report on double click
Show detail window on double click
Show hypertext window on double click
-
Report to
  Display
  Printer
  File...
Printer setup...
Printer font...
-
Enhanced display output
Dull gray windows
-
Hypertext options...

Tools

Save data for Super Project Analyzer...
Super Project Analyzer
-
Project Printer

Project Analyzer was written in Visual Basic 4.0
by Tuomas Salste, Helsinki, Finland
© 1994-1997

Visit the Visual Basic Shop WWW page
http://www.netgate.net/~vbshop/vb.html
vbshop@netgate.net
Shareware Visual Basic tools

List files

Product Analyzer can produce list of files that belong to your project. To get a list, choose **Report|List files**. A dialog box will appear.

You can get lists sorted by:
- File name
- File type

If you want just a list of file names, leave the **Details** box unchecked. If you check it, the list will contain the following additional data:
- Comments
- Which files need procedures in this file
- Which files this file needs

Note: You get the commented lines in the (declarations) section of your .BAS, .FRM, .TXT or .GLB file. The program shows those comment lines that come before the first non-comment line, like this:

```
' This is a function module
' These comments will be reported

Dim Text As String
' This comment will not be reported
```

The file list goes to the output device defined with the **Output goes to** radio buttons.

List procedures

Product Analyzer can produce list of procedures that belong to your project. To get a list, click the **List procedures** button or choose **Report|List procedures**. A dialog box will appear.

You can select if you want the following data included in the procedure list or not:
- Comments
- Which procedures use this procedure
- Which procedures this procedure uses

In the registered version there are also options to list procedures that are or are not needed by your program. This is useful for finding **dead code**.

Note on comments: The program will show those commented lines at the beginning of each sub/function that come before the first empty or non-comment line. Commented lines before and after the sub/function declaration line are shown, as well as procedure description (VB 4.0), if available. An example:

```
' This sub was created by N.N.
SUB MySub (Byval x As Integer)
Attribute VB_Description=Prints x + 5 on the form
' This sub does the following:
' It takes the parameter x and ...

' This line is not shown any more
Form1.Print x + 5

END SUB
```

The procedure list goes to the output device defined with the **Output goes to** radio buttons.

Archive Project Files

Project Analyzer helps backing up your program files. Currently it supports two archiver programs, namely **ARJ** (current version: 2.41a) and **PKZIP** (version at this time: 2.04g). You can also easily build file lists of your project files in order to manually copy/archive them without effort.

Pack Project Files Dialog

Select files to be packed (or files whose file name are to be saved in the list file) in the list box. By default, all VB files belonging to your project are selected, but no DLLs. These are the files that you most probably want to save.

If you want to add files to the list you can click the Add button. Files ending in **.lst** are considered to be list files that contain a list of more files you'd want to add to the list box. You can create these .lst files with any text editor or with Project Analyzer itself.

The **ARJ Command** and **PKZIP Command** fields contain the commands and command switches used to create new archive files.

Sample ARJ Command: ARJ a MYARCH.ARJ !MYPROJ.LST
Sample PKZIP Command: PKZIP MYARCH.ZIP @MYPROJ.LST

You only have to supply the blue part of the command line; Project Analyzer does the rest. Now you can pack by pressing one of the **Pack** buttons.

By default, a temporary file is used to tell the archiver program what files to pack. If you want to save that list file for later use you can click the **Select** button.

Having named a list file, you can save the list file only if you want. In other words, a new button name **List files only** will appear.

Introduction

Project Analyzer is a Finnish shareware tool designed to help creating, maintaining and reporting applications developed in Visual Basic 3.0 and 4.0. The program should work with VB 2.0 too.

Project Analyzer makes a full, two-phase source code analysis. It collects detailed information about all the files, procedures, controls, variables, constants, classes etc. in the VB project. The results can be viewed on the screen, in textual format in your favourite word processor, or on paper.

What is needed to run Project Analyzer

Project Analyzer needs Visual Basic 4.0 DLLs to run. You can run it with Visual Basic 3.0, but you will then have the DLLs instead. Visit my WWW site (**http://www.netgate.net/~vbshop/vb.html**) to download the needed DLLs.

Project Analyzer is available in both 16-bit and 32-bit versions. **You only need one version** for analyzing VB 3.0, VB 4.0 (16-bit) and VB 4.0 (32-bit) projects. If you are only using VB 3.0, get the 16-bit version of Project Analyzer and the required 16-bit DLLs.

Features

Full source code analysis
- File details: procedure and control lists, file level cross-references, lines of code, comment to code ratio and other metrics
- Procedure details: cyclomatic complexity, lines of code, and other metrics, procedure level cross-references
- Find dead and live procedures, variables and constants. Remove dead constants.
- Variable and constant declaration information, both dead and live
- Call trees
- Project metrics like cyclomatic complexity, comment to code ratio, ...

Reports for maintaining the project
- Sub and Function lists with comments for reporting purposes
- File lists with comments
- DLL reports
- Problem report
- Project summary report
- Variable and constant lists with dead or live variables & constants
- Design quality report
- Needless globals report
- Name shadowing report

Other powerful features
- Project file archiving with PKZIP or ARJ
- Hypertext style code viewer
- FRX file view
- Removing unused constants from CONSTANT.TXT and DATACONS.TXT

See version history for new features.

Why? Why???

Maintain phase costs rule the software industry. When developing applications, especially those that someone else has to maintain, it's important to keep track on what each sub/function does, and what other procedures they need to work.

This is where Project Analyzer can help you. With it you can produce detailed cross-reference reports with supplied comments. You can view call trees. And you can find dead code, dead variables, and dead constants, and various other problems too. You can get the documents on the screen, on paper, or even on your company Intranet as HTML reports.

Maintaining programs is **hell**. But Project Analyzer will turn it into **heaven** :-)

How to use it?

Using Project Analyzer is simple. Open a **.mak** or **.vbp** file with the Open Project command in the File menu and wait. Project Analyzer will collect information about the **.bas**, **.frm**, **.frx, .cls,** and **.dll** files in the background.

Note: If you are using VB 3.0, remember to save all your files in text format.

The analysis consist of two phases. The first phase collects the most data, and the second one will check for cross-references and find dead code etc. If a menu command or a button is greyed, don't panic, just wait for the analysis to end.

Hint: Test Project Analyzer by opening the included TEST30.MAK or TEST40.VBP file (named after VB30 and VB40). These are pure demo projects for you to see how Project Analyzer works. You can of course use your own projects too for this purpose.

When the analysis has ended, just double click a procedure or a file to see the results. Remember to push the **Hypertext** button. Don't forget to examine the View and Report menus either, many of the most powerful features reside there. Feel free to test your own projects too - Project Analyzer will not modify them!

Registration
Collecting information
Back to Contents

Collecting Project Information

When you start Project Analyzer, or when you choose Open Project... from the File menu, you get a dialog box asking you to choose a project file (a VB **.mak** or **.vbp** file).

Project Analyzer examines the project file to find out what files belong to your project. After determining the type of each file, the program further looks inside these files.

What information Project Analyzer searches for in the files:

Project Analyzer examines the **.bas, .frm, .frx, .cls, .dll** and other files belonging to your project. As an exception, Project Analyzer doesn't analyze .vbx, .ocx nor .res files.

If you use VB 3.0, remember to save the files in text format. Project Analyzer can't read VB files saved in binary format, other than the .frx files. It will analyze .dll files based on the declarations in your basic code. - If the program finds a binary file it will simply ignore its contents. This will result in incomplete analysis, like showing dead variables that aren't dead in reality.

Project Analyzer builds a list of all files and procedures (i.e. subs and functions) that your project consists of. Code is analyzed at form and procedure level to list the controls and the variables & constants. Furthermore, procedure and variable & constant cross-references are examined.

Note: Information is collected in the background. So, when Project Analyzer is examining your files you can do other things like running other programs - or you can view the information the program has collected so far. There are some things you can't do when the collecting is in progress, like getting a list of all procedure calls - that's because the program hasn't yet finished collecting that piece of information!

The program reads your files twice. You'll notify when it's ready by looking for the word "Ready!" at the lower left corner of the program window. Should you want to stop the process, you can press the Stop button. To speed up the collecting process, the program checks if you have pressed the Stop button only every now and then, so be patient!

Use Option Explicit. You will get better results if you're declaring all your variables. This is good practice anyway.

Report Formats

Project Analyzer has very powerful, formatted report options. You can get the reports
a) On the screen
b) On paper
c) On the Clipboard
d) In a file for your favourite word processor
e) In a help file
f) On your company Intranet as HTML files

Use the **Options|Report to** menu command or one of the **Report to** radio buttons to select the report format.
Depending on what you select, you either get a formatted report, or a text-only version.

1. To the display
This is the default. The output goes to a special Display window where you can read the text. You have 2 options to take:
a) A formatted report
b) An unformatted, plain text report (from which you can copy parts to the clipboard)
The type of report you get is determined by the **Options|Enhanced display output** menu option.

There are 3 of these Display windows available. You can use them side-by-side.

2. To the printer
If you want, you can choose the output to go to the selected Windows default printer. This report is formatted.

To select a printer, click **Options|Printer setup**. To choose a printer font, click **Options|Printer font**.

3. To a file
Project Analyzer can direct its output to:
a) a Windows text file (plain text)
b) a MS Windows Write .wri file (plain text)
c) a Rich Text Format (RTF) file (formatted report)
d) an RTF file ready for Help Compiler to turn the report into a Help file (formatted report)
e) an Internet Hypertext (HTM) file (formatted report)

You can even tell Project Analyzer to run an editor, browser or the Help Compiler to open or process the report after it has been created. To do this, just check the **Run browser/editor after report** check box in the Save report dialog box.

Running a browser/editor/Help Compiler

Project Analyzer needs to know which program to run after creating a report. By default, the Windows default associations are used. Press the right arrow button in the Save report dialog box to see which program will be launched.

Possible choices for editors/browsers/Help Compiler:
a) TXT: Windows Notepad or any other text editor
b) WRI: Windows Write or WordPad
c) RTF: Microsoft Word or Windows WordPad, or any other word processor supporting RTF
d) RTF for Help Compiler: The Microsoft Help Compiler (HC.EXE, HC31.EXE, HCP.EXE).

HC.EXE comes with Visual Basic, and is in your \VB\HC directory. After running the Help Compiler, Project Analyzer starts WinHelp.
e) HTML: Netscape or Explorer, or any other WWW browser

You can change the default by writing your own command line, or by association the file name extension (**.txt**, **.wri**, **.rtf**, or **.htm**) with your browser/editor. In Win 3.x, you do this from **winfile.exe | File menu | Associate**, and in Win 95, from **Control Panel | View menu | Options | File types**.

Multiple reports in one file

You can add multiple reports in one file by selecting an existing file. Works for text, RTF, help and HTML files. If you have selected RTF for Help Compiler, Project Analyzer will also create a Table of Contents page that shows all the reports in that file.

Note: When producing large reports (over 32kB in size) you can get an "Out of string space" error when the output is going to the unformatted display window, or to a Write file. The report will be truncated to 32 kB. The solution is to use the enhanced display, printer, text file, RTF file, or HTML file option, which allow reports of any length.

About Project Analyzer

Contact information - Future versions - Registration

Contact me!

I'm eager to get any comments about Project Analyzer. I'd like to get your comments even if you don't register and even if you didn't like Project Analyzer at all. My Internet email address is **vbshop@netgate.net.**

You may also want to visit my VBShop WWW page **http://www.netgate.net/~vbshop/vb.html** Product information will be available through that page. You will also find information about my other shareware and freeware products there.

New versions

Project Analyzer is still under further development. New versions with further enhancements can be expected in the future. New features will mainly be available in the registered version only.

I will notify all registered users about any new versions. Registered users will get the new versions for free. Distribution of the updates is done solely on the Internet, usually by WWW.

Registration

The following text applies to **unregistered** users only.

Project Analyzer is **shareware.** If you use this program for anything else than evaluation purposes, you'll need to pay a small registration fee. When you pay the fee, you get the full, registered version of Project Analyzer with more features. You will also get **all future versions for free**. **Getting new versions for free requires an Internet email address.**

Be sure the read the following:
1. The general registration instructions in REGISTRA.HLP. They apply to all my Visual Basic tools.
2. The special registration instructions below. They apply especially to Project Analyzer.

Price

The price for registering a single copy of Project Analyzer 3.1 is:

USD	80	USA
FIM	350	Finland

You can also register in your own currency, but please email to negotiate the exchange rate.

Multiuser licences

1 user	$80
Additional users	$50 / user (2 to 4 users)
Additional users	$40 / user (5 or more users)

Here are the prices for 1 to 10 users:

Users	Total price
1	$80
2	$130
3	$180
4	$230
5	$270
6	$310
7	$350
8	$390
9	$430
10	$470
Additional users	$40 / user

Add-ins

The following add-ins require a separate registration. In addition, they need the registered Project Analyzer to work.

Add-in	Single registration
Super Project Analyzer	$20
Project Printer	$20

Source code

The source code is also available for an extra $100. This price includes all updates. You don't have to purchase several licences for the source, but you need to have registered Project Analyzer.

Payment method

The possible payment methods are described in REGISTRA.HLP.

Registration keyword

After registering, you will get a registration keyword that will enable the registered features. Input this keyword in the About box when Project Analyzer starts, and you will be registered.

The same applies to Super Project Analyzer, but you input the keyword when Super Project Analyzer starts.

Other utilities for Visual Basic:

- DBtoVB Wizard - automatically generate functions for creatíng a new database - no coding!
- DB Lock - examine the behaviour and locking scheme of your multi-user database application
- DB Structure - print the structure of an MS Access database
- InputBox - enhanced InputBox function for VB

See the Visual Basic shop WWW page (**http://www.netgate.net/~vbshop/vb.html**) or email to me for more information!

Hypertext

(version 1.1 onwards, registered version only)

You can see your code in hypertext form (like text in this help file) by using Project Analyzer's hypertext feature. The hypertext window is a read-only code window but with additional features. You can see called procedures easily by clicking at the underlined procedure names. If a constant or variable is underlined, you can jump to the declarations section where it was declared.

To open the hypertext window, select a file or a procedure in the **Files** or **Procedures** list box and either press the **Hypertext** button or select **View|Hypertext** hypertext in the menus. Another method is to press the **Control** key down and then click in the **Files** or **Procedures** list box. Still another method is to select Options|Show hypertext on double click.

To see the procedures that call the shown procedure click the **Called by** button.

Hint: Try right-clicking highlighted variables, constants and procedures. You will get a menu with links to their definitions, and in the case of global and module-level definitions you will also see where they are referenced and assigned to!

Options
You can change the colors of different types of hotlinks by selecting **View|Options**. If you want, Project Analyzer can **~~overstrike~~** dead variables too.

Note: This feature is available in the registered version only. In the unregistered version you can only use it with small projects.

Project Printer can save your source code in hypertext format into a **.hlp** file. The result is quite similar to the Hypertext window, but you no longer need Project Analyzer to view the file.

List variables and constants

Product Analyzer can produce list of the global and module-level variables and constants that belong to your project. To get a list, choose **Report|List variables and constants** in the main window, or press the **Report** button in the Variables and constants window. A dialog box will appear.

You can get two kinds of lists:
1. List all **global** variables and constants in alphabetical order
2. List all **global** and **module level** variables and constants ordered by file they appear in

The file list goes to the output device defined with the Output goes to radio buttons.

Note 1: Variables and constants information is available when analysis phase 1 is complete.

Note 2: You can get more versatile variable and constant lists from the Variables and constants window.

Version History

See a more detailed version history in VERSION.TXT.

Version 3.1 (Spring 1997):
- Super Project Analyzer
- Project Printer
- Project metrics
- Name shadowing report
- HTML reports
- Help file reports
- Lots of enhancements and bug fixes

Version 3.0 (Summer-Autumn 1996):
- Added support for Visual Basic 4.0
- Project Analyzer now needs Visual Basic 4.0 DLLs to run. If you only use VB 3.0, please visit VBShop (**http://www.netgate.net/~vbshop/vb.html**) to download the required DLLs.
- Class modules supported
- Hypertext window enhanced
- Call tree enhanced
- Needless globals report
- Variables with no type in Problem report
- A lot of enhancements to various features

Version 2.1 (September 1995):
- Project summary report
- Generate constants module command
- Modifications to look better with Windows 95
- Enhancements in the hypertext window
- Average cyclomatic complexity (version 2.1.05 onwards, registered version only)
- About 25% faster analysis (appears in the second phase) due to improving PROJECT.DLL
- Formatted reports on the screen too (version 2.1.07 onwards)
- DLL Report (version 2.1.07 onwards, registered version only)
- Can handle larger projects than the previous versions

Version 2.0 (August 1995):
- The Display window is no more the only way of seeing the results of an analysis. Separate detail windows have been implemented.
- File details and Procedure details can now be viewed on the screen in a separate window in addition to the reports
- File details show the form's icon and controls too
- New Variables and constants window with information about references to them (dead variables & constants, referencing files)
- New Call tree window
- File level call trees added in addition to procedure level call trees
- FRX files can be examined too (just double click the FRX file in the main window)
- Shows declared functions in external DLLs
- Problem report (registered version only)
- Formatted reports using RTF files
- New statistics, like cyclomatic complexity (registered version only) and lines of code for each procedure
- Find command now in the Hypertext and Procedure details window too

Version 1.1 (March 1995):
- Find dead procedures (registered version only)
- Global and module level Variable and constant lists

- Procedural call trees (registered version only)
- Hypertext style code viewer (registered version only)
- Saving the results of the analysis for future use (registered version only)
- A **Find procedure** command to quickly locate a procedure

Version 1.0 (February 1995):
The initial release.

Call tree

(version 1.1 onwards, registered version only)

You can see the procedure and file dependencies in the form of a call tree. The tree is available in two formats:

1. Call tree in a separate window

Use the **View|Procedure call tree** and **View|File call tree** menu commands to see an expandable view of the dependencies. You can get a textual report of the tree by pressing the **Report** button.

2. Textual report

The reports are available using the **Report|Procedure call tree** and **Report|File call tree** menu commands. Three subtypes of a procedure call tree report are available:
2.1. All procedures
2.2. Procedures in selected file only
2.3. The selected procedure only

If a procedure calls itself or another procedure that then eventually call the first procedure back, it will be marked **<recursive call>** in the tree.

Note 1: This call tree features are available in the registered version only. In the unregistered version you can see the call tree only for small projects.

Note 2: The call trees are available after analysis phase 2 has completed.

Note 3: You can see dependencies in hypertext or list form too.

Note 4: You may get an out of memory error when trying to see large call trees in the dedicated window, or the program may hang. In this case, get the textual report instead. It is recommended not to take the All procedures call tree on anything but small projects. On large projects, the best choice is the Select procedure only call tree.

If you don't like the background colors I've chosen for Project Analyzer, you can set them to gray using **Options|Dull gray windows**.

File details window

(version 2.0 onwards)

The **File details** window shows information about a specified file in your project. The easiest way to see File details is to double click the **Files** list box in the main window.

File details can show information about:
- Form files
- FRX files
- Basic modules
- Class modules
- DLL's

Basic files (FRM, BAS, CLS)

- Code lines
- Size in kilobytes
- Form icon (FRM files only)
- File version (FRM files only): The file format of the FRM file. Note that VB 3.0 still creates FRM version 2.0 files
- Subs and functions: Name, type (functions only), lines of code, and 'Dead' status if the procedure is not used by your project.   - Double click a procedure to see procedure details.
- Variables and constants defined in this file: Global and module-level definitions.
- Controls (FRM files only)
- List of files that call this file (BAS files only, after phase 2)
- List of files that this file calls (after phase 2)
- Various metrics (after phase 2)

Some of the information will not be available before the analysis phase 2 has been completed. Question marks '?' indicate that the analysis is not complete yet. To see the results after completion of analysis use the View|Refresh menu command, or press Ctrl-R.

DLL files (DLL, EXE, ...)

DLL's are analyzed like BAS files, except:
- Variables and constants are not listed
- Files that are called by this DLL are not analyzed
- The lines of code in a DLL cannot be analyzed
- DLL file version is not analyzed

Only those procedures that were declared with a Declare statement are shown.

FRX files

Project Analyzer can examine the contents of a FRX file that is there if you saved your FRM file in text format and the form contained pictures. The FRX file details window shows a list of the pictures that are in the FRX file, and the names of the controls these pictures are for.

Report and Hypertext

To see the procedure details as a textual listing, press the **Report** button. To see the selected procedure as hypertext, select the **View|Hypertext** menu command.

Note: VBX, OCX and RES files are analyzed by Project Analyzer.

Procedure details window

(version 2.0 onwards)

The **Procedure details** window shows information about a specified sub or function. The procedure can be a basic procedure, an event or a library procedure.

The following information is shown:
- Procedure name and type
- Procedure declaration
- Procedure file
- The 'dead' state of the procedure (after phase 2)
- Lines of code (doesn't apply to library procedures)
- Procedures that call this procedure
- Procedures that are called by this procedure (doesn't apply to library procedures)
- Various metrics (after phase 2)

Some information is available only after analysis phase 2 has been completed. Question marks '?' indicate that the analysis is not complete yet. To see the results after completion of analysis use the View|Refresh menu command, or press Ctrl-R.

To see the information as a textual listing, press the **Report** button. To see the entire procedure code, press the **Hypertext** button.

Variables and constants window

Project Analyzer can show a list of all global and module-level definitions either as a textual report or in a list box. To see the list box, either press the **Vars & consts** button in the main window, or select the **View|Variable and constant list** menu command.

You can choose from the following options:
- Show constants
- Show variables
- Show global definitions defined in the (declarations) section of a module
- Show module-level definitions defined in the (declarations) section of a form or module
- Sort by file
- Sort by name
- Show live definitions (those that are referenced somewhere in the program)
- Show dead definitions. Those ones are dead that are not referenced even once in the program, even though a dead variable may be assigned to.

To see the actual definition of a variable or a constant, press the **View declaration** button. That will bring up the Hypertext window with the corresponding (declarations) section. To view a textual listing, press the **Report** button.

To see which procedures reference or assign to the selected variable or constant, select the **View references** command. A list will show up with all assignments and references to that variable or constant.

Note 1: Variables and constants information is available when analysis phase 1 is complete. The reference statistics are available when phase 2 is complete. To display the information, close the window and reopen it after the analysis.

Note 2: To see procedure-level variables or constants, use the Hypertext window.

Problem report

(version 2.0 onwards, registered version only)

Project Analyzer can provide you with a **Problem report** about your project. It lists some minor but sometimes annoying things.
- Basic files saved as binary - save as text to work with Project Analyzer **and** for better reliability with VB.
- Files without Option Explicit. It is good programming practice to always declare your variables.
- Dead procedures, i.e. subs and functions that are not in use
- Variables without a specified type (implicit variants)
- Too complex procedures
- Minimizable forms without icon
- Forms with ControlBox but without icon (needed in Win95)
- EXE file without title and icon (this happens if you haven't made an EXE file yet)

It is recommended that you take the Problem report only after the analysis has been completed.

Project metrics

(version 3.1 onwards, registered version only)

To monitor your programming performance, programmers often use some simple metric, like lines of code or EXE file size. However, this is not always enough. Project Analyzer helps you to monitor the understandability, complexity, and reusability of your code.

You can see various metrics in many different ways
- **View|Module metrics** and **View|Procedure metrics** windows are good to find long and complex procedures and modules
- Project summary report is good to see the overall performance
- Design quality analysis report is also good to see the overall performance
- File details window shows metrics by file
- Procedure details window shows metrics by procedure

Below you can read a brief discussion on understandability, complexity and reliability, and the various metrics used to estimate them.

Understandability

Bad understandability will most probably result in more errors and maintaining problems. You can estimate the understandability of your program with the following metrics:

Lines of code / procedure (View|Metrics, Design quality report)
Procedures longer than 50 lines are often too long.

Comment to code ratio and **whitespace to code ratio** (Design quality report)
The more comments in your code, the easier it is to read - and understand. Whitespace is also important for legibility.

Length of identifiers (Design quality report)
The longer your variable, procedure etc. names are, the more likely they are to be descriptive.

Cyclomatic complexity (View|Metrics)
Target for less than 10. See below.

Depth of conditional nesting (see View|Metrics)
Target for less than 5. See below.

Complexity

There are many kinds of program complexity: structural, informational, and logical, for example. High complexity may result in bad understandability and more errors. Complex procedures also need more time to develop and more testing.

The best measures for estimating the complexity of a procedure are:
- Lines of code
- Cyclomatic complexity
- Informational complexity
- Structural fan-out
See below for more information.

Reusability

Reusability is a magic word in programming. Measures for reusability are
- Reuse ratios reported by Super Project Analyzer (reuse in a group of projects)

- Structural fan-in. If it's high, the procedure/module is called (reused!) many times.
- Number of class modules

What do the different metrics mean?

Note: You can see most of the metrics only after the whole analysis has been completed.

Lines of code

As simple as it may seem, lines of code is quite a good measure of how complex a program is. Project Analyzer calculates lines of code as follows:

Lines of code = Total lines - Commented lines - Empty lines

Lines of code does not include control declarations in a .frm file.

Cyclomatic complexity

(McCabe)

Cyclomatic complexity is a measure of the structural complexity of a procedure. The higher the number, the more complex the procedure, and the harder it is to maintain it. Cyclomatic complexity for VB procedures is calculated as follows:

Cyclomatic complexity = Number of Branches + 1

Branches are caused by IF, SELECT CASE, DO...LOOP and WHILE...WEND statements.

"Normal" values for cyclomatic complexity range from 1 (very simple) to 9 (moderately complex). If C is more than 10, you may want to split the procedure.

Cyclomatic complexity is the minimum number of test cases you must have to execute every statement in your procedure. This is important in testing. Carefully test procedures with the highest cyclomatic complexity values.

An average cyclomatic complexity, as well as the distribution of complexity, is reported in the Project Summary Report.

Note: Cyclomatic complexity often gives quite high values for procedures with long SELECT CASE statements.

Nested conditionals

Nested conditionals metric is related to cyclomatic complexity. Whereas cyclomatic complexity deals with the absolute number of branches, nested conditionals is only interested in how deeply nested these branches are.

If you have a procedure with deeply nested conditionals, you should consider splitting it up. Those procedures can be very hard to understand and error-prone.

Nested loops

Nested loops is a very rough estimate of the mathematical complexity of a procedure. The more nested loops there are in a procedure, the more likely it is that those loops take up a significant amout of time to execute.

Structural fan-in/fan-out

(Constantine & Yourdon)

For procedures:

Structural fan-in = the number of procedures that use this procedure

Structural fan-out = the number of procedures this procedure calls

For modules:

Structural fan-in = the number of modules that use variables, constants, or procedures in this module

Structural fan-out = the number of modules whose variables, constants, or procedures this module needs

A high structural fan-in denotes reusable code.

The higher the structural fan-out, the more dependent that procedure is on other procedures, and more complex too.

Informational fan-in/fan-out and informational complexity

(Henry & Kafura)

Lines of code, cyclomatic complexity, or structural fan-out are not perfect in predicting the "real" complexity of a procedure. For example, a procedure may access a number of global variables and be very complex without having to call many other procedures.

Informational fan-in = Procedures called + parameters referenced + global variables referenced
Informational fan-in estimates the information a procedure reads

Informational fan-out = Procedures that call this procedure + [ByRef] parameters assigned to + global variables assigned to
Informational fan-out estimates the information a procedure returns

Combined, these give a new metric: *informational fan-in x fan-out*. This is reportedly good in predicting the effort needed for implementing a procedure, but not so good in predicting complexity. To predict complexity, we need a new metric: informational complexity. It is calculated as follows:

Informational complexity = lines of code x (informational fan-in x informational fan-out)

Rich Text Format

(version 2.0 onwards)

Project Analyzer can produce formatted reports and save them in a Rich Text Format (RTF) file. A number of popular word processors read RTF files, including Microsoft Word and Windows 95 WordPad. Project Analyzer also produces RTF for the Help Compiler to turn reports into **.hlp** files.

More information on report formats

MS Word

A special option for Microsoft Word users is the **Create Microsoft Word style RTF** check box on the **Save Report** dialog box. This option may produce ugly reports for other word processors (for example, for WordPad).

If this option is set, Project Analyzer optimizes the RTF reports for Word. It uses styles like Normal, Heading 1, Heading 2, ... You can easily create a Table of Contents for your report with Word.

RTF for Help Compiler

You can also create an RTF file for the Microsoft Help Compiler. This way, you can turn the reports into Help files. Project Analyzer will create the necessary .prj file and launch Help Compiler automatically.

The **Create Microsoft Word style RTF** checkbox has no effect on RTF for Help Compiler.

Options
(version 2.0 onwards)

1) Show report on double click
2) Show detail window on double click
3) Show hypertext on double click

These options let you specify what happens when you double click the **file list** or the **procedure list** on the main screen.

1. Show report on double click option is for compatibility with early versions of Project Analyzer (1.0 and 1.1), and will produce the standard reports for a file or a procedure.

2. Show detail window on double click option lets you easily view some file details or procedure details.

3. Show hypertext on double click option lets you access the hypertext window easily.

Features in the registered version

The following features are available in the registered version only. Some of them will work in the shareware version too, provided your project has no more than 10 source files.
- Deadcode information
- Dead variable and constant information
- Generate constants module command (for VB 3.0)
- Project metrics for professional programming
- Call trees
- Problem report
- DLL report
- Name shadowing report
- Needless globals report
- Hypertext window

Registration information

Add-ins

Project Analyzer has a couple of add-ins that need a separate registration.

- Super Project Analyzer is an add-in tool for analyzing groups of applications.
- Project Printer prints source code and generates hypertext source documents.

Project summary

(version 2.1 onwards)

To get a summary of your project, use the **Report|Project summary** command. This report includes some statistics about your project. It also shows some Visual Basic limitations, like the global symbol table size of your project, and its maximum size, 64 kB.

Not all of the measures are exact, these approximated measures are marked as *(approx).*

Generate constants module

(version 2.1 onwards, VB 3.0 only, registered version only)

The use of the CONSTANT.TXT and DATACONS.TXT files in a project tends to take up a large amount of memory, because these files include so many **global const** declarations, and only a few of them are needed in one project. Including a large number of unused constants in a project will fill up the global symbol table, as well as decrease the performance of the program.

If you use Project Analyzer, you can solve the problem easily. When you are developing your application, you can include the CONSTANT.TXT and DATACONS.TXT files in your project; you don't need to cut and paste any declarations to another global module. When your project is about to be ready, just analyze it with Project Analyzer, and select **File|Generate constants module**. This command will generate a new module including only those constant declarations that are needed by your application.

After generating a new module, remove the old one (CONSTANT or DATACONS) from your project and add the new one. You're done.

Note 1: This command applies mainly to VB 3.0 projects. VB 4.0 doesn't require you to declare a vast amount of constants, but has predefined constants instead.
Note 2: This command is available only after analysis phase 2 has completed.

DLL report

(version 2.1.07 onwards, registered version only)

This report lists all DLL files used by the project. All declared DLL procedures are listed, and if they are not used, they are marked as "dead".

Use this report to examine which DLLs you should distribute along with your application, and to remove unnecessary DLL declarations.

Nested loops

(version 3.0 onwards)

Select **View|Nested loops** to see the procedures that have nested **For..Next, Do..Loop** or **While..Wend** loops.

Generally speaking, loops are the parts in a program that take the most CPU time. When analyzing the efficiency of different algorithms, most emphasis is put on analyzing nested loops and what is inside them. With Project Analyzer, you can easily find those procedures that have nested loops.

Super Project Analyzer

(version 3.1 onwards, needs registered Project Analyzer and a separate registration)

Super Project Analyzer is an add-in tool for Project Analyzer. It is registered separately ($20). In addition to that, it needs the registered Project Analyzer to work.

A Super Project is a group of ordinary Visual Basic projects that share some files. Super Project Analyzer analyzes these project groups and reports which procedures, variables and constants etc. are shared or dead.

Instructions

To use Super Project Analyzer, save a couple of normal .vbp/.mak analyses using the **Tools|Save data for Super Project Analyzer** menu command. This will write a .sud (Super Project Data) file for each VB project. This file includes information on the modules, procedures, variables and constants used by that project.

After saving a few (at least 2) .sud files, start Super Project Analyzer (**Tools|Super Project Analyzer**). Then open the .sud files with the **Project|Add project** command in Super Project Analyzer. You can now see the modules, procedures, variables and constants used by all of these projects.

Note: In the unregistered Project Analyzer, Super Project Analyzer can only handle projects with less than 10 source files. If you have registered Project Analyzer, Super Project Analyzer can handle up to 25 source files per project without separate registration. The registered Super Project Analyzer, of course, has no such limitations.

Features

Select **Shared only** to limit the display to those elements that are included in at least 2 projects. **Shared Only** may also show dead elements that are included in at least 2 project but are not used by them.

Select **Dead only** to see which elements are not used by any project. You may consider removing them.

Note: Super Project Analyzer does not list VBX, OCX, or FRX files. It does list DLLs and DLL procedures.

Reports

To get a report on what you see on the screen, press the **Report** button, or select the **Report|Report what you see** menu command**.**

Select the **Report|Reuse report** menu command to get a report on reuse ratios. The report includes reuse ratios for modules, procedures and variables & constants by project and for the whole super project as well.

Reuse ratio for modules = Shared modules / All modules in project

Reuse ratio for procedures = Shared procedures / All procedures used in project
(thus excluding dead ones)

Reuse ratio for vars&consts = Shared vars&consts / All vars&consts references in project
(thus excluding ones that are completely dead or that are only assigned to)

Saving a Super Project definition

To save a Super Project definition to a .sup file, use the **File|Save Super Project** menu command. You can later open this file to skip opening each project separately.

Note: In addition to the .sup file, you will need to have the saved .sud files in order for the analysis to work. The .sup file only includes the file names of those .sud files.

Registration

To use Super Project Analyzer, you will need to register both Project Analyzer and Super Project Analyzer. Super Project Analyzer registration fee is USD 20.

More about registration

Project Printer

(version 3.1 onwards, needs registered Project Analyzer and a separate registration)

Project Printer is an add-in tool for Project Analyzer. It is registered separately ($20). In addition to that, it needs the registered Project Analyzer to work.

Project Printer is designed to document VB source code. It can
- Print your source code
- Document references
- Generate hyperlinked source documents in a HTML or HLP file

Project Printer is not just an ordinary code printer. It
- highlights variables, constants and procedure calls
- overstrikes dead variables and constants
- can list calls to and from a procedure
- can report various metrics along with the code
- can generate Table of Contents
- even generates hyperlinked documents (like in the Hypertext window)

You can start Project Printer by selecting the **Tools|Project Printer** menu command.

Note 1: Project Printer gives best results after the analysis has been fully completed.

Note 2: The unregistered Project Printer can only handle projects with up to 10 source files. The registered Project Printer has no such limitations.

Output types

Project Printer is versatile. In step 2/3, it asks you to select the output type. Do this in the main window in the ordinary way, as would do when printing any other report.

Paper. Ordinary formatted source code printout with Table of Contents. Optionally with metrics and 'called by' list.

Help file. The easy way to browse a project. Project Printer generates a help file that is very much similar to the Hypertext window of Project Analyzer. Calls are hyperlinked. The Help file is always there for you to take a quick look at your project, even on computers without Project Analyzer.

HTML file for Intranet documentation. Calls are hyperlinked. Useful for a group of programmers. Even store your code bank in HTML.

RTF file. Format the report in any way you want with your favourite word processor.

Other types. You can get the report in any other report type Project Analyzer supports, for example, on the screen, but the results are not as usable as in the above types.

Options

Project Printer has a number of options in step 3/3. Default values are set based upon which report type has been selected when you continue from step 2/3.

Source style. **Simple** is a fast, plain text report. **Enhanced** parses through the code and highlights identifiers, overstrikes dead ones. In HTML and HLP, it also sets hyperlinks for references.

Show contents. On paper, generates a Table of Contents with page numbers at the end of the report. In a Help file, it's on a separate page. In other report types, it is at the start of the report.

List global/module-level vars and consts. Check this if you want a list of vars/consts declared in (declarations). The list includes information on where a variable or constant has been referenced or assigned to, or if it's dead.

Show incoming calls. Lists the procedures that call this procedure.

Show outgoing calls. Lists the procedures that this procedure calls.

Show metrics. Shows various programming metrics, like complexity etc.

Registration

To use Project Printer, you will need to register both Project Analyzer and Project Printer. Project Printer registration fee is USD 20.

More about registration