GLIDER: Simulation Language
WINDOWS Users Quick Guide
Win95/98/NT
IEAC/FACES - CESIMO/FAI

Project CDCHT I-524-95-02-AA
Universidad de Los Andes. Mérida, Venezuela.

May 2001

GLIDER Development Group

Contents

GLIDER WINDOWS version

This is a quick helping guide for users of the *WINDOWS* version of the *GLIDER* Simulation Language [1]. This version requires the previous installation of *BORLAND Delphi* version 4 or 5. *BORLAND Delphi* [5] was also the development tool.

The *Graph* Unit from *TURBO Pascal* [4] language, in which the original *MSDOS* version of GLIDER is based, is emulated in this version.

To install GLIDER unzip the distribution file dGLIDER.zip (or dGLIDER5.zip for Delphi 5) into hard drive. Make a shortcut to the executable WinGli.exe, change the icon to the icon file WinGli.ico.

Quick Guide

Double click for execution upon the shortcut *WinGli*. Or click the *Start* Button, select *Run* and enter the command:
<drive>:\DGLIDER\WinGli.

GLIDER programs must have extension `.gld`. Select the program name with the options *File*|*Open* in the Main Menu of WinGli. Select *Compile* to compile the program. The compilation process opens two Console type windows in sequence. Ignore any message of the type “File not Found”, press any key when asked for and close manually the last window in case it does not close automatically. If the compilation was successful, a corresponding `.exe` file must have been created. To run the executable select *Run* in the Main Menu. Or run it outside WinGli as any other WINDOWS executable.

The option *Complete Compilation* is the default selection in the menu *Options* of the Main Menu. If there were GLIDER compilation errors you can edit the program with the WinGli editor. If there were *Delphi* compilation errors or in case you can not find the problem in your *gld* file, change the Compilation mode to *Partial Compilation* in the *Options* menu and compile again. With this compilation mode selected, the Glider compiler only generates the *ObjectPascal* files (*<program>.dpr, Pru.pas and Unid1.pas (Unid2...)*). Use your Delphi installation to edit the project, using the options *File*|*Open* to open the corresponding `.dpr` project that was created by WinGli. You can compile and run the project directly in the Delphi IDE, after adding the *dGLIDER* directory to the Search path list (*Project*|*Options*|*Directories*|*Conditional Search path*)

Creating and editing a GLIDER program

Users of the WINDOWS version have to create and edit their programs using any WINDOWS or MSDOS text editor program, as Notepad or MS Word (saving the file as ASCII text with extension .gld) or using the *WinGli* editor. GLIDER syntax is case *insensitive*. Refer to the *GLIDER Reference Manual* for language syntax and model programming examples. A web version of that Manual can be seen in http://www.faces.ula.ve/~carlosd/gliweb

GLIDER program file names must have extension **gld**, as myexample.gld

---

GLIDER Vocabulary

Pascal Keywords

and, array, begin, case, const, div, do, downto, else, end, for, function, goto, if, in, label, mod, not,
of, or, procedure, record, repeat, set, then, to, type, until, var, while, xor

---

Pascal Types

boolean, byte, char, double, file, integer, longint, pointer, procedure, real, string, text, word

---

GLIDER Keywords

DBTABLES, DECL, end., FACTORS, GFUNCTIONS, INIT, MESSAGES, MODULE,
NETWORK, NODES, PROCEDURES, RESPONSES, STATISTICS, TABLES

GLIDER Types

CONT, FREQ, GPOINTER, RET, STRING, str80, DOEVENT, DONODE

GLIDER Instructions

ACT, AGRE, ASSEMBLE, ASSI, BEGINSCAN, BLOCK, CLRSTAT, COUNT, CREATE, DBUPDATE, DEACT, DEBLOCK, DISASSEMBLE, DISFEL, ENDSIMUL, EXTFEL, EXTR, FIFO, FREE, GRAPH, INTI, LIFO, LOAD, MENU, METHOD, MOVE, NOTFREE, ORDER, OUTG, PAUSE, PREEMPTION, PUTFEL, REL, REPLIC, RETARD, SCAN, SELECT, SENDTO, SORT, STAT, STOPSCAN, SYNCHRONIZE, TAB, TRACE, TRANS, UPDATE, UNLOAD, UNTRACE

GLIDER Functions

ber, beta, bin, DMEDL, DMSTL, ENTR, erlg, expo, gamma, gauss, LL, lognorm, max, maxi, MAXL, MEDL, min, mini, MINL, modul, MSTL, norm, PFIRST, PLAST, poisson, RANDOM, rnd, TFREE, tria, truncate, unif, unifi, VOIDFEL, weibull

---

Pascal Functions and Procedures

abs, append, arctan, assign, blockread, blockwrite, close, CloseGraph, copy, cos, delay, dispose, eof, exp, filepos, filesize, gettime, getdate, int, length, ln, new, open, ord, OutTextXY, read, readkey, readln, release, reset, rewrite, round, seek, sin, sqrt, str, trunc, upcase, val, write, writeln

---

Colors

BLACK, BLUE, BROWN, CYAN, DARKGRAY, GREEN, LIGHTBLUE, LIGHTCYAN, LIGHTGRAY, LIGHTGREEN, LIGHTMAGENTA, LIGHTRED, MAGENTA, RED, WHITE, YELLOW

Compiling a GLIDER program

Complete Compilation

Use the program *WinGli* to compile a GLIDER program, for example **myexample.gld,** selecting its name with the *File*|*Open* option in the main menu.

In case that GLIDER Compiler finds any syntax error, an error message is given, indicating line and column numbers where the error was detected and the compilation process ends. Use your text editor for correcting the errors and compile again until successful. If you can not find the error cause, please, contact us via e-mail to the address shown at the end of this section.

If no errors were detected, in the program folder, besides the program source file, the executable program with the same name but with extension .exe, must be also there. For example, after selecting the *Compile* option, the Windows Explorer will show the names **myexample.gld** and **myexample.exe**.

The GLIDER precompiler must generate correct Delphi ObjectPascal programs, so the Delphi compilation must not give any error message. If any error message happens to come and you cannot find the problem, please, send us an e-mail reporting the case and include if possible the source program to one of the following addresses:

sananes@faces.ula.ve

Partial Compilation

If you have any concern about the Delphi ObjectPascal modules that the GLIDER precompiler generates, select the *Partial Compilation* mode in the *Options* menu in Main Menu. In this mode, WinGLI. Only generates the ObjectPascal modules and leaves them in the program folder. Open the following generated files in the Delphi IDE:

<name-prog>.dpr
pru.pas

unid1.pas

If the GLIDER program defines one or more additional modules (**MODULE** instructions), files with names unid<*x*>.pas will also be generated for each additional module, *x*=2,3....

In the Delphi IDE you can compile the corresponding project (***<name-prog>***.dpr file) using the option Project/Build in the Delphi main menu that completes the process producing the executable file <*name-prog*>.exe. Prior to Delphi compilation you have to add the *dGLIDER* directory to the Search path list (*Project*|*Options*|*Directories*|*Conditional Search path*) in the Delphi IDE.

The user is responsible for deleting the intermediate files.

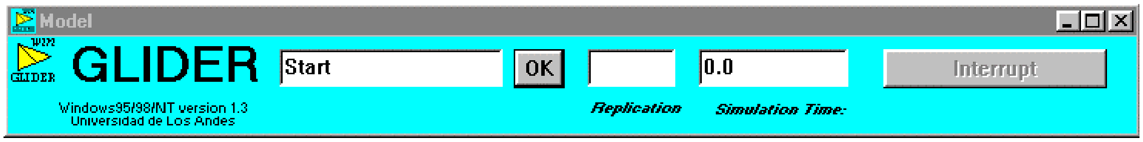
Console Application

If your program contains any read instruction from standard Input or write to standard Output, select *Console Application* in the *Options* menu in Main Menu. Otherwise, the program will raise a Exception condition when attempting to execute the first read from or write to Console.

Running a GLIDER program

After successful compilation you can run the program. Enter the program name as a command, with the *Run* option of the Windows *Start* Button or Open Windows Explorer and double click in the program name (*Model* in this example) or select the option *Run* in the main Menu of WinGli.

GLIDER programs start presenting the Control window:



During execution follow the directives given in the *Message edit box* -the first one to the left-and then press the next *OK* button. Use the *Interrupt* button if you want to interrupt normal execution and go into *Trace* mode.

For details about menu presentation, operation and run options, consult the GLIDER Language Reference Manual [3]

[Note: During Trace mode you can interrupt again to go to the main menu and then to end the run. Trying to return to normal mode is not working properly**]**

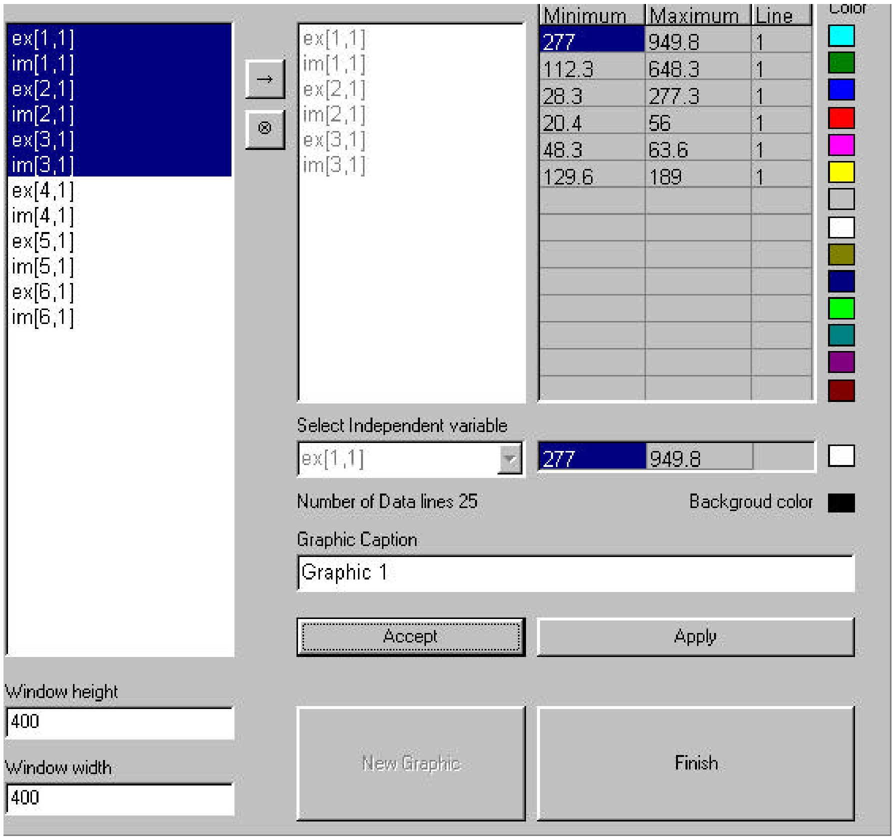
Running programs with graphics

GRAPH Instructions

Programs may have up to 100 **GRAPH** instructions, each one displays a graphic window. A standard graphic window size is assigned depending on the graphic type. For time graphics, the standard window width is as wide as the width of the screen. But you can customize the graphic windows size by placing assignment instruction for *both* predefined variables: **WindowWidth** and **WindowHeight** in the **Init** section of a GLIDER program. Example.:

```
WindowWidth:=400;  WindowHeight:=400;
```

When the simulation run ends, you can choose the option *Close Graphics* in the main menu. The closing process grants focus to each graphic window in creation reverse order. For each activated window, the system places a *Continue* message in the Control window and then waits until the user press the next *OK* button. In the meanwhile, the user can save the graphic to a bitmap graphic selecting the *File* option in the graphic window menu. If any variable data value were outside the graphic limits, the system shows a warning with the message 'Graphic Scale was changed'



ex[1,1]
im[1,1]
ex[2,1]
im[2,1]
ex[3,1]
im[3,1]
ex[4,1]
im[4,1]
ex[5,1]
im[5,1]
ex[6,1]
im[6,1]

ex[1,1]
im[1,1]
ex[2,1]
im[2,1]
ex[3,1]
im[3,1]

Minimum	Maximum	Line	Color
277	949.8	1	🟦
112.3	648.3	1	🟩
28.3	277.3	1	🟦
20.4	56	1	🟥
48.3	63.6	1	🟪
129.6	189	1	🟨

Select Independent variable

ex[1,1]

277	949.8	□

Number of Data lines 25

Backgroud color ■

Graphic Caption

Graphic 1

Window height

400

Window width

400

and then the graphic is redraw using the system changed scales.

The closing graphics process is also executed when the user choose the *Quit* option.

FILE Instructions

Files generated by **FILE** instructions in a program can be seen and saved choosing the *Graphic* option in the main menu after the end of the simulation run. An *Open dialog* form allows to selected the **FILE** generated file with variables data. **[**A File exception error will occur is the file doesn't have the expected **FILE** format, and the program aborts**]**. Then, the *Graphic dialog* form appears:

First of all, a set of independent variable (no more than 14) has to be selected at once from the left list box, using the Windows modes (contiguous pressing the Shift key or dispersed pressing the Crtl key). Then use the arrow button to place them in the right list box. From the right list box you can drop unwanted variables using the ⊗ button.

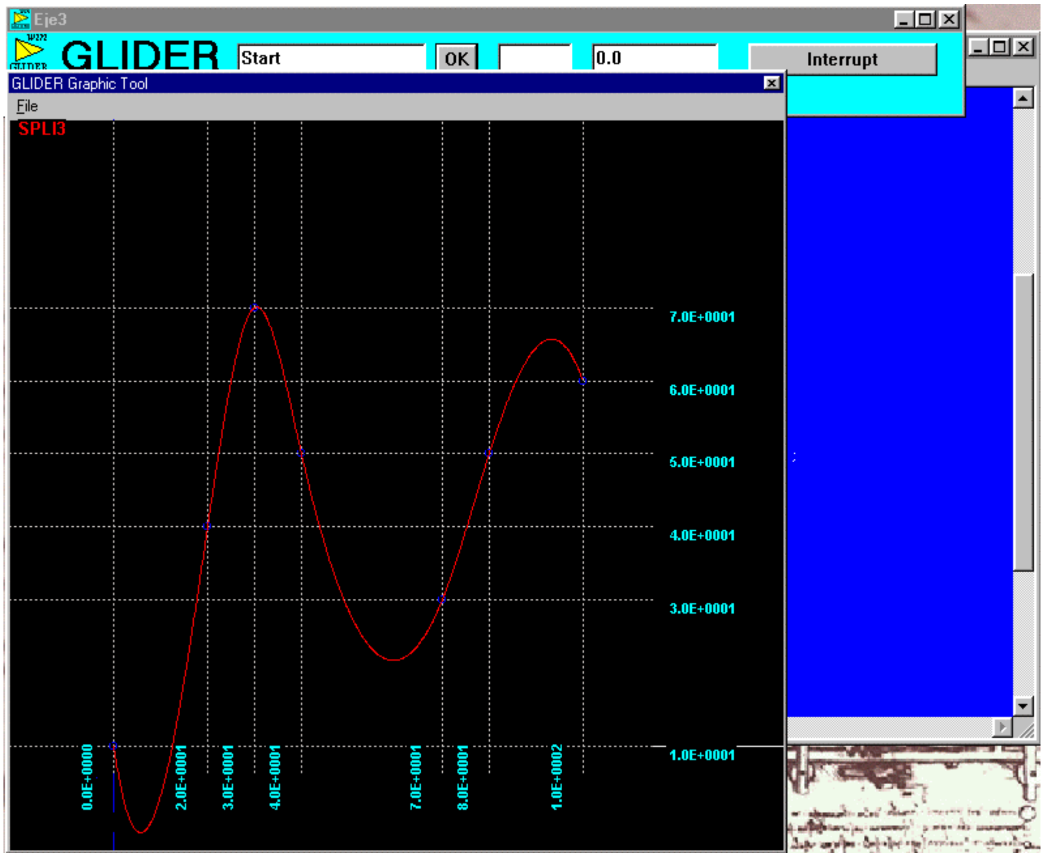
The independent variable may be reselected from the combo list bellow. The graphic caption may be changed as well. After pressing the *Accept* button, the system displays configuration data in the rightmost table, including pre-selected colors for the dependent variables chosen. Similar data for the independent variable are displayed bellow. The user can change all those values at will. Color may be changed by clicking in the colored rectangle.
Window size (height and width) may be changed before or after accepting the variable selection.
When the *Apply* button is pressed, the graphic window is shown with the size, caption, scales and color chosen. At that point, the user may choose to define a new graphic from the same file, by pressing the now enabled button *New Graphic* or quit the graphic dialog, pressing the *Finish* button.

After quitting the Graphic dialog, all defined graphics are displayed and then the message Closing Graphics appear in the *Message edit box*. The system performs the closing process, as described in the previous section (***Graph Instructions***).

When the Main Menu comes back, you can select the *Graphic* option again to define others graphics from another or the same **FILE** generated file.

INTI Graph instructions

The GLIDER instruction INTI when applied to a GFUNCTIONS identifier, generates an interactive graphic window. Using the mouse buttons, the user can add or modify points to the function. The redefined function replaces the original program definition for the current simulation. On exiting the graphic window, the redefined function may be saved to a file for posterior use.



In either case, GRAPH or FILE generated graphics, if there is enough room in the screen, the system tries to arrange the graphic windows in tile mode, otherwise in cascade mode.

Examples of GLIDER programs

A collection of examples of GLIDER programs is available in the **DGLIDER/DEMOS** Folder.

The file names of the examples have the form **EXE**<*x*>**.gld**, *x* = 1,2,3,..20

Differences of implementation with MSDOS version

Some characteristics of GLIDER have been implemented differently from the MSDOS version or have not been fully tested. Some are new to the Windows version.

Differences on implementation are:

- Modes of operation: TP version offers two modes of operation: Normal and Trace. In both modes the program is interruptible. In the WINDOWS version there are the same modes, the program is interruptible using the *Interrupt* button in the control window.

  But in the Windows version the trace is always recorded in a file. When the program ends or the user stops the tracing, the file content is shown in a text window, if it fits. If the file is too large, as usually is, a warning message pops and then the user is asked to save the file and see it with the WordPad editor or similar.

- Delphi ObjectPascal requires every for-loop-control variable to be a local variable to the file being compiled. In the Windows version for-loop-control variables used in node codes must be declared local to that node, using the syntax:

<node-name> (<type>) [<list-of-successors> ,] var <var-declarations> ::

- The predefined Delphi ObjectPascal files *Input* and *Output* are implemented following the standard Pascal definition, without the TurboPascal facilities, like *GoToXY* or *ClrScr*, included in the *Crt* Unit that Delphi does not support. Programs that write to *Output* or read from *Input* have to be compiled with the option *Console Application* checked in the *Option*s menu of WinGli (*WinGLIDER*) main Menu. This option allows the program to open a console window for use of files *Input* and *Output*. Working directly in the Delphi IDE, this options is activated using the main menu option *Project/Options/Linker/Generate Console Application* or placing the special commentary `{$APPTYPE CONSOLE}` after the initial program line in the .dpr file (main file of Delphi projects) .

- First parameter in instruction **FILE** may be a variable name or file name constant within single quotes.

- The *database* features have not been tested in the WINDOWS version.

New features in Windows version:

- **USES** instruction: A GLIDER program may contain *one* **USES** instruction, if so, it must be the first instruction in the **DECL** section. Same syntax as Pascal. This feature allows the Delphi compiler to compile programs that include types, variables and routines defined in external Units, thought the GLIDER precompiler does not recognizes them. To avoid GLIDER compilation errors, the next second feature was added
- Marking instructions: Placing a '#' character as the first character of a line tells the GLIDER precompiler to ignore that line and pass the rest of it as it is to the generated ObjectPascal file. All instruction lines that contain references to identifiers belonging to external Units must be marked. The user is responsible for the proper syntax, otherwise, the Delphi compiler will report compilation errors. You can correct them using the GLIDER IDE or the Delphi IDE.
- Delphi *Dialogs* **Unit**: The *Dialogs* **Unit** is always included in the system generated **USES** list. The windowed communication routines that this Unit contain, like *ShowMessage, InputBox*, etc., may be used, but in lines marked with '#' to avoid GLIDER compilation errors.
- *WDGRAPH* **Unit** is also always included. *WDGRAPH* is the Windows Delphi emulation of the *GRAPH* TurboPascal Unit, with some additions [6]. It was developed for the GLIDER Windows version, but it may be distributed independently by the GLIDER Development Group.
- Multiple Graphics: GLIDER Windows version support multiple **GRAPH** instructions, each one generating a graphic window for display. Graphics may be saved in BMP format.

References

1 Domingo C., Hernández M., Sananes M, Tonella G.: *Lenguaje de Simulación GLIDER: Guía de Referencia* CESIMO-IEAC, Universidad de Los Andes, Mérida, Venezuela, 1994.

2 GLIDER Development Group: *GLIDER Reference Manual* IEAC-CESIMO/ULA, Venezuela & Interdisciplinary Research Center, Department of Mathematical Sciences, SDSU, August, 1996.

3 Grupo de Desarrollo GLIDER. *Guía de Usuarios RedULA*. IEAC/FACES - CESIMO/FAI, Universidad de Los Andes, 1999.

Financiamiento: CDCHT-ULA.

4 BORLAND International, Inc.: *Turbo Pascal version 6.0*

5 Inprise BORLAND Delphi, versions 4 and 5

6 Sananes, M., Sayago, F. *Transporte de Programas por Emulación de Bibliotecas. (Parte 2: Caso GLIDER a Plataforma Windows 95/98/NT).* IEAC/FACES, CESIMO/FAI. Presentado en las II Jornadas de Estudios Estadísticos. Abril 1999. Financiamiento: CDCHT-ULA.

GLIDER

Carlos Domingo, Marisela Hernández (Diseño lenguaje y factibilidad*) Renato del Canto, Cristina Zoltan* (Asesores) *Carlos Domingo, José G. Silva, Giorgio Tonella* (Diseño lenguage extendido) *Carlos Domingo* (Desarrollo) *Marta Sananes, José G. Silva, Giorgio Tonella* (Asesores) Cruz Acosta, Herbert Hoeger, Luis Fuentes, Tania Jiménez, Francisco Palm, Segundo Quiroz, Oswaldo Terán, Kay Tucci (Colaboradores) **Versión UNIX/LINUX*:*** *Carlos Domingo, Marta Sananes* (Desarrollo) Milton Quero, Niandry Moreno (Colaboradores). **Librería Emulación TPEX:** *Marta Sananes, Rafael Tineo* (Desarrollo) Solveig Maldonado (Colaborador**). Versión Windows:** *Carlos Domingo, Marta Sananes, Fernando Sayago, Francisco Palm.*

GLIDER: Simulation Language
WINDOWS User Quick Guide
IEAC/FACES - CESIMO/FAI
Financiamiento: Consejo de Desarrollo Científico y Humanístico CDCHT
Universidad de Los Andes. Mérida, Venezuela

May 2001.