GiD-NASTRAN Interface: Guideline for development of a high quality interface

Aleix Valls*†

*International Center for Numerical Methods in Engineering (CIMNE)
Universidad Politécnica de Cataluña.
Campus Norte UPC, 08034 Barcelona, Spain.
E-mail: aleix@cimne.upc.es

†Compass Ingeniería y Sistemas S.A.
C/Manuel Girona 61 Bajos, 08034 Barcelona, Spain.
E-mail:aleix@compassis.com – Web page: www.compassis.com

Keywords: NASTRAN, GiD, interface, problem type, TKWidget, Tcl-Tk, customisation, RamDebugger.

Abstract. *The GiD-NASTRAN Interface is a high quality interface between GiD (the personal pre and postprocessor) and NASTRAN, a well-know FE package. This paper discusses general aspects of a high quality interface development, focusing on the different development steps, such as interface structure definition, timing, level of adaptation, Tcl-Tk customisation, and the use of the available tools and resources for the development. Furthermore the GiD-NASTRAN Interface will be presented as model solution for the development of a professional interface.*

INTRODUTION

The objective of this article is to describe the process of creation of a commercial interface between GiD and a program of simulation. The functionality of the interface is to communicate the program of pre and post GiD process and a program of numerical simulation, in this case NASTRAN, based on the Finite Elements Method. Without to enter in technical details of the solutions adopted for GiD-NASTRAN interface, the article presents some of last available options in the creation of a commercial interface.

We understand for commercial product an application designed for the professional use of third people. Consequently, compression of the handling and the reliability of the application developed are basic and fundamental properties of the product.

GiD-NASTRAN Interface offers an environment of work that permits to the user the complete preparation of model FEM since the initial state (CAD data) to the final state of post-processing of results. Thus it is possible to assign and to define materials and contour

conditions, generation of complex meshes, visualization of the results with powerful graphical tools and finally as innovation also it is possible to import the NASTRAN model, that is to say, material, conditions and general information of the model also are imported and not only information about the NASTRAN mesh. See Figure 1.

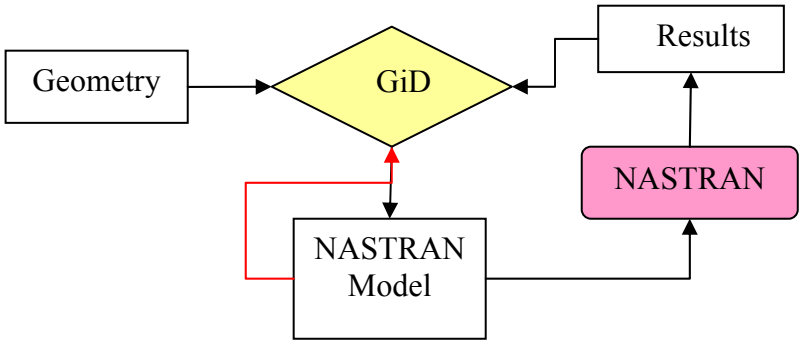

Figure 1 Flow of information of the interface

For more information about GiD-NASTRAN Interface look at references [i,ii].

LEVEL OF ADAPTATION

GiD offers the possibility of creating rapidly graphical interfaces for user (GUI) by means of the files of configuration *.mat, *cnd, *.prb. These GUIs are of enough quality for those conditions or simple properties of the model, for example, prescribed temperatures or materials with simple properties,…For other conditions or more complex materials properties (change of the properties according to the temperature, interrelationship between data,…) the functionality that GiD provides to us for the generation of GUIs by means of the use of configuration files is limited. In this case the migration to a more powerful technology is necessary: The GiD extension based on the language TCL-TK [iii].

The process of development of a GUI by means of the language TCL-TK is subdivided in three stages:

1) Creation of the classic files of configuration of GiD *.mat, *cnd, *.prb. with those fields that we want to replace with graphical elements TK in state "hidden", being able to be all of them.
2) Development of the GUI and its functionality in TCL-TK in independent form to GID and its data structures.
3) Communication of the GUI with GiD and its internal data structures, by means of the facility TKWidget [iv,vi]. The facility TKWidget allows to communicate the information got in the GUI with internal data of GiD.

This way of proceeding in independent stages has demonstrated being the most efficient provided. The modifications and location of errors in the TCL-TK script generator of the GUI are realized more comfortable and rapid form before the third stage. The stage of communication of the GUI with GiD is the most complex of the process. Provided that the control of flow of information is difficult to monitor, the use of a tool of treatment of errors (debugger tool) it is of great help in this stage. There exist several programs for debug errors

on TCL-TK scripts on the market, but RamDebugger[v] is with difference the one better that adjusts to the needs of this stage of connection. In the next section of this article there are described the characteristics of this tool and other utilities of support to the development of interfaces.

A characteristic to stand out of this process of progress of the GUI of the interface, it is the capacity of reutilization of previous developments of the interface. This idea answers to the following situation:

Let's suppose that we have developed an interface for GiD using only the classic configuration files. We have besides the fact that the file (*.bas) that GiD uses to write the input file for the program of simulation continues the structure of the configuration files. The writing of this file is a task that can become long and complex according to the quantity and complexity of the input file for the code of simulation. Like example the file *.bas of GiD-NASTRAN interface has more than 5000 lines. Let's imagine that we want to update the interface using TCL-TK. Does it mean that all the work realized in advance is useless? By no means, we can re-take advantage of the whole development of the previous interface, only it will be necessary to hide the fields of the configuration files preserving their structure and to connect them by means of TKWidget to the news GUI developed in TCL-TK. Thus the file *.bas will continue being functional.

In Figure 2 appears the evolution the GUI for materials of GiD-NASTRAN Interface. The left image corresponds to the GUI generated with the configuration files of GiD, the image of the right hand corresponds to the GUI developed by means of TCL-TK and the facility TKWidget, as it is possible to observe the functionality and handling of the GUI has experienced a drastic change.

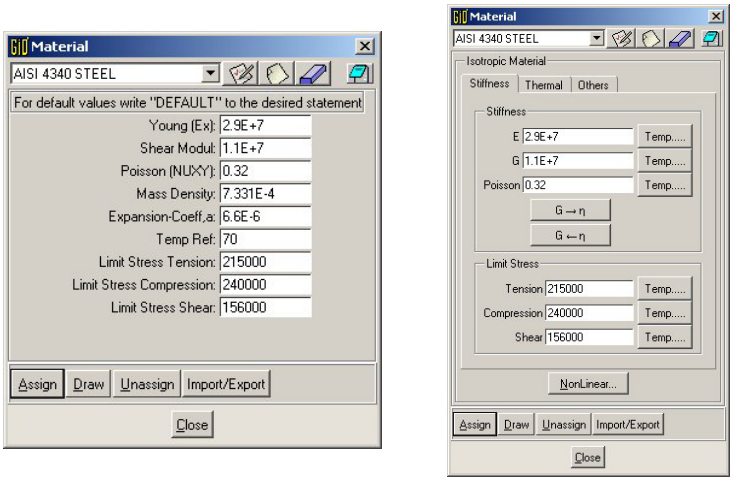

Figure 2 Evolution of Material GUI. GiD-NASTRAN Interface

The use of TCL-TK like extension language of GiD takes more possibilities as the modification of menus, access to the internal structures of information of GiD on meshes, geometric entities… The possibilities of adjustment of GiD using the extension in TCL-TK endow to this application of an entire versatility, capable of being able to adjust to a wide spectrum of needs.

A last comment about the use of the extension TCL-TK, it is its application in the creation of the input file (*.bas) for the program of simulation. Since it is wise the file *.bas allows to call to procedures TCL by means of the *tcl instruction. With the use of this option can appear the extreme case to use only TCL for the development of the file *.bas. During the development of GiD-NASTRAN interface it has been proved that this option is not ideal provided that the proprietary language of GiD used to write the file *.bas is faster than the use of TCL commands. This due to the fact that the own instructions of GiD are optimized in its programming in C ++ does that the ideal option computationally speaking, is the mixed use of two languages with priority to the proprietary language of GiD and the use of TCL restricted to those where the functionality of the GiD language is not sufficient.

AVALIBLE DEVELOPMENT UTILITIES

In this section there appears a compendium of the resources and available applications for the development of interfaces for GiD.

The most important application of support to the development of interface for GiD is RamDebbuger [v]. RamDebbuger is a graphical debugger for the scripting language Tcl-TK.

With RamDebugger, it is possible to make **Local Debugging**, where the debugger starts the program to be debugged and **Remote debugging**, where the program to debug is already started and RamDebugger connects to it.

RamDebugger has additional capabilities like:

- Editing the code. It is possible to edit the code inside its own editor and resend the new code without closing the debugged program.
- The TCL-TK source code is colorized and supports automatic indentation.
- When stopping the debugger is one source code line, it is possible to view all the variables and expression values, as well as change them.

These characteristics do of RamDebugger a tool of great help for the development of scripts in TCL-TK. To standing out, the characteristic that transfers major utility for the development of interfaces for GiD, the possibility of being able to debug errors of scripts that are executed externally. It is possible to debug the errors of a script executed by GiD remotely with RamDebugger and to be able to control the real flow of information between the GUI and GiD. With this functionality the task earlier mentioned of connecting the GUI to GiD is simplified considerably and is not realized so blindly. Like example the use of this tool during the development of GiD-NASTRAN Interface facilitated enormously the development of piece of news GUI in substitution of the classic ones, as well as the implementation of new facility for the user as the control of errors in the assigned conditions, automatic calculation of moments of inertia, graphs of stage of values,… Another element to stand out is the reduction of time in the development of the scripts basically due to the rapid location of errors.

Other interesting options of RamDebugger are the automatic coloration of the files configuration of GiD. This option probably of minor utility in files *.cnd, *.mat, it has been demonstrated of great utility for the file *.bas, given the limitations of indentation and location of blocks of code.

In Figure 3 and Figure 4 there appear some of the GUI developed for GiD-NASTRAN interface using the tool RamDebugger

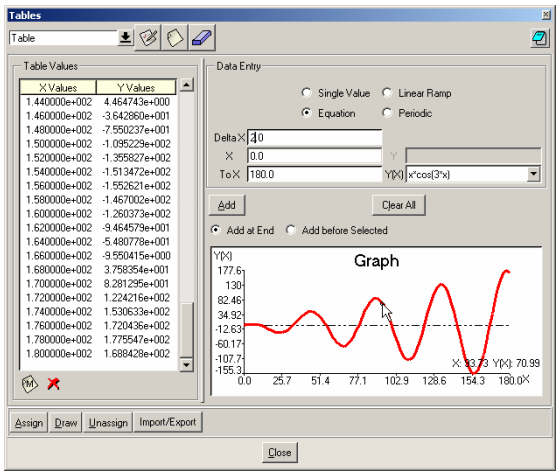

Figure 3 GUI Table of values

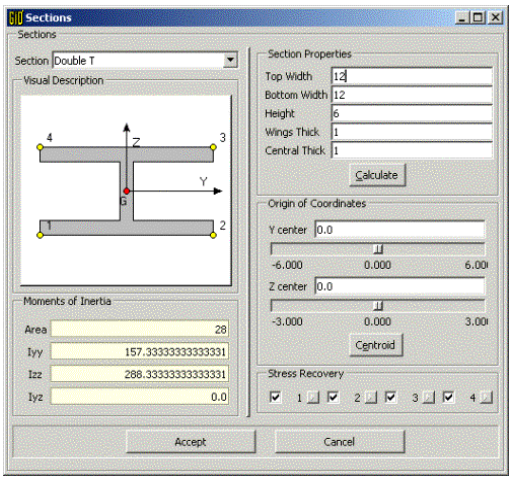

Figure 4 Sections definition

TIMING

The time of development of an interface using TCL-TK is bigger than the time necessary for the development of a classic interface.

The curve of learning of the grammar of the configuration files of GiD is of soft earring, that is to say, in a little time it is possible to develop an operative basic interface. On the other hand, in use of the TCL-TK extension the time of learning of the language TCL-TK will be much major. Still not to be necessary a very deep knowledge of the language, it has been proved that the introduction to the TCL-TK language is not trivial and asks from time. In GiD-NASTRAN Interface it has been decided in favour of a mixed development of the interface, in the first stage of development of a new functionality one uses the classic options of GiD, later it is considered to be its progress in TCL-TK. This way of proceed is showed efficiently in several aspects as reliability of the final product, rapidity in of the development and division of the work in different teams of development.

REFERENCES

[i] Compass Ing. y Sistemas S.A. , *GiD-NASTRAN Interface User Manual*, Compass Ing. y Sistemas S.A, (2002).

[ii] Compass Ing. y Sistemas S.A., *GiD-NASTRAN Interface Reference Manual*, Compass Ing. y Sistemas S.A, (2002).

[iii] TCL-TK Web page http://www.tcl.tk/software/tcltk/.

[iv] CIMNE, Centro Internacional de Métodos Numéricos en Ingeniería, *GiD Reference Manual,* CIMNE (2000). http://gid.cimne.upc.es

[v] RamDebugger Web page http://gid.cimne.upc.es/RamDebugger/RamDebugger_toc.html

[vi] J. Perez Suit, *Advanced Problem Type Development: data dependencies (DEPENDENCIES) and GUI customization (TKWIDGET)*, 2nd Congress on Advances and Applications of GiD, Congress proceedings CIMNE (2004).