Using Tk as remote GUI frontend for 4GL-database applications

Volker Schubert
Brueckner & Jarosch Ing. GmbH
Erfurt, Germany

Using Tk as remote GUI frontend for 4GL-database applications

Volker Schubert
Brueckner&Jarosch Ing.GmbH
Erfurt, Germany, 99084
leo@bj-ig.de

Abstract

This is a short report about our experience using Tk as GUI frontend. We wrote a compiler called **F4GL** *, source compatible to the* **INFORMIX 4GL** *database language. (4GL means 4th generation language) This language was originally designed to create UNIX database applications with text-terminal output. 4GL programs compiled with* **F4GL** *generate Tcl-commands as graphic output (the text output is also still available). These Tcl-commands are sent over TCP/IP to a presentation server which will provide for conversation into graphic objects. We have presentation servers for X11 and Windows.*

4GL Server *is the presentation server for all Windows platforms and is implemented using Tcl/Tk. It is an Internet-Server like lpd or rshd whose services are available at a specific TCP port. We found different solutions for the network connection to the remote GUI / presentation server and had to solve various problems, which are discussed in detail in this poster. Writing a internet server in Tcl which evaluates Tcl-commands is very easy, but it introduces a security hole. The machine running* **4GL Server** *is accessible from the internet with tcl-commands. That's why we implemented a special protocol to authenticate the users of the* **4GL Server** .

*We made some extensions, some of it appears to be useful for the whole Tcl-community. Until now we maintained our own Windows-port of Tk3.6, because the actual Windows-Tk4.x releases from Sun still have limitations, we will explain the reasons and show some performance bottlenecks (***http://www.bj-ig.de/speed.html***). In the near future we want to switch to official releases because the Tk4.xx and Tk8.xx versions offer many advantages, especially safe interpreters and the Tcl-Plugin. A good overview about the* **F4GL** *can be found at* **http://www.4js.com/f4gldoc.html***, this is the webpage of* **4JS***, our trade-partner.*

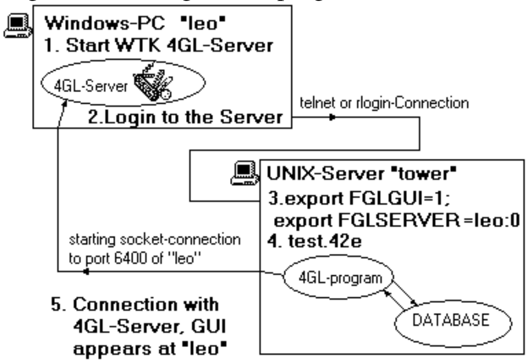
1 History

Two years ago we searched for a solution to migrate existing 4GL-database-programs to GUI's. First, we discussed a cross-compiler to translate 4GL into another database-language with builtin GUI (New Era, Powerbuilder, Gupta a.s.o) but this would exclude expierenced 4GL-programmers from the development. Therefore we decided to write a source compatible compiler.That produces programs with a configurable, flexible GUI-frontend. We choose Tk for the remote GUI, because we had some experience with it (usaging it for visualization in measurement tasks under Linux/X11), and it was free. The only problem was a good Windows port for the actual Tk3.6 version, because the majority of customers were expected using Windows (especially in Germany). We took one 16bit-port from **Software Research Associates Inc.** and started development at the Win32s platform to support native controls, sockets, optimize speed, and eliminate bugs. We call this Tk derivative **WTK** and it's of course freely available with all sources. See more information at **http://www.bj-ig.de/wtk.html**. Of course we want to switch now to the main distribution, to use all the very nice features of Tcl/Tk8.0, but there is one reason , which doesn't allow us to use it: performance.

2 Usage/Invocation

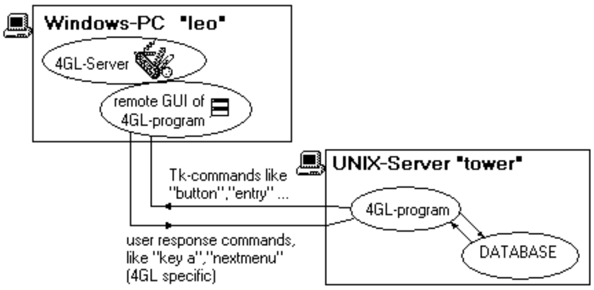
We'll begin with a short overview, of how to launch a database program, and how the communication between db-program and remote GUI is established (using Windows as frontend system). The user clicks on an icon, and a `rlogin` connection to the UNIX database host is established, the command line is automatically transferred, and the 4GL-program starts.
It then reverse connects to the Windows-machine (it reads an environment- variable `FGLSERVER` comparable with the `DISPLAY` variable for X11-applications) with our **4GL Server** at a specified port (6400) and transferres Tcl/Tk-commands like `button` or `entry` to the windows-machine. These commands are evaluated and

Figure 1: Starting a 4GL-program at the UNIX-Server



shape the GUI. Because the `rlogin` window can be kept hidden, the user images locally the program started.

Figure 2: Communication between 4GL-program and remote GUI



If the user clicks on a button or writes to an entry, special bindings guarantee that this action is sent back over the socket as a simple text command.
The `rlogin` connections and the socket-connections of the 4GL-programs run in one application at the windows side, this is our product **4GL Server** ,completely written in Tcl (**WTK** =Tk3.6+some extensions).

3 Extensions

According to the frontend-character of the used widgets we made some extensions to Tk3.6. Because the runtime system of the database application controls all activities of the remote GUI, some widgets used have totally changed bindings as mentioned above.

3.1 The class-patch

The most important patch for us is the "class-patch" which allows the Tk-developer to give every widget another class, not only the `frame` and `toplevel` widget. This allows for example to create entries with different classes, each class can have other definitions of options in the option database. For each class there are other bindings possible. Both possibilities are often used in **4GL Server** . We already posted the patch to comp.lang.tcl, but it's unfortunately not contained in the actual distribution (one reason for writing the poster).

3.2 native controls

The main reason for developing **WTK** was to have a native look and feel of the GUI at the Windows-platform. The following controls are native windows controls under **WTK** : all kinds of `button`'s,`label`,`scrollbar`,`menu`,`term`. .

3.3 wtcp extension

wtcp-tcl is yet another socket implementation for Windows, it supports all kinds of tcp-sockets, asynchronous callbacks(WSAAsyncSelect) and winsock specific calls . Now the "socket"-command is available in the major distribution, and we want to switch, but not all functionality we need for **4GL Server** is included(setsockopt, transfer binary zero's for protocol- handling).

3.4 term extension

`term` is a terminal widget with configurable 3D pseudographic symbols and configurable attributes. For example, the attribute bold you can assign a 3D-background color and a relief (comparable with Tun Terminal Emulation). `term` was used to implement a full-featured xterm Emulation in tcl, which is part of **4GL Server** . In the middle of the year we will introduce it for the tcl-community as a TK8.0-extension, but until now it's **WTK** -specific and only as native Windows widget available.

Figure 3: the 4GL-program D4 with tcl-output