The Mensurasoft-System: Software Drivers for Scientific Devices in the Most Common Computer Languages on All Major Operating Systems with a Prototype API for Measurement Data Processing Applications - Practical Examples

P. Dieumegard *

Independent Researcher, Orléans, France

Abstract

For years now, although many scientific devices have a port for computer connection, the diversity of various connection types results in lack of software able to use the totality of them. This paper presents how to: (1) build software drivers with a prototype Application Programming Interface API for scientific devices using dynamic libraries under various combinations of connection ports, programming languages and operating systems, (2) employ these drivers in software programs capable of managing more than one device at the same time, and (3) use two specific software implementations of this whole Mensura (Measurement) System to select appropriate built driver, define communication settings, and collect-store-process-present experimental data.

Keywords

Mensurasoft, measurement, devices, dynamic library, programming languages, MS-Windows, Linux, driver

Introduction

In this paper, "scientific devices" are mainly measurement devices: pHmeters, voltmeters, thermometers etc, but sometimes they are actuators: motors, heaters, voltage-generators etc. Many scientific devices can be connected to a computer; in the past, usually through an ISA, PCI or RS-232, but nowadays via USB too.

Sometimes there is software to use them, often there is no software. When a software is sold or given with a device, it can be used only with this device or similar devices from the same sup-

plier. This causes problems to professional scientists, as well to teachers of sciences, and to simple amateurs of sciences. Can we imagine programs able to manage all of them?

This problem is similar to that of office software: how to make programs (spreadsheets, word-processors etc) able to manage all printers - not only already existing printers, but also the printers which will appear in the next years. The solution is similar; by using device drivers. There is a printer driver for each printer, and we can make a driver for each measurement driver.

However, there are two differences between office programs and scientific programs. Scientific devices are more various than printers, and usually produced in small quantities. Scientist often must make himself/herself the program for his/her aim.

In the present work we will show how to use "scientific drivers" from the system Mensurasoft named by the Latin word mensura which means measure. Fig. 1 gives an illustration of the Mensurasoft system architecture: separation between programming for devices and programming for humans is an obvious necessity.

The necessity of Drivers

The construction of scientific devices is not easy. Making a pHmeter is a job for specialists, with special glasses for the electrode, special electronics, correction for temperature, etc. In the same way, building a precise scale to measure weight, or an accurate lux-meter are works of specialists. These specialists can supply a program for their devices, but not for others.

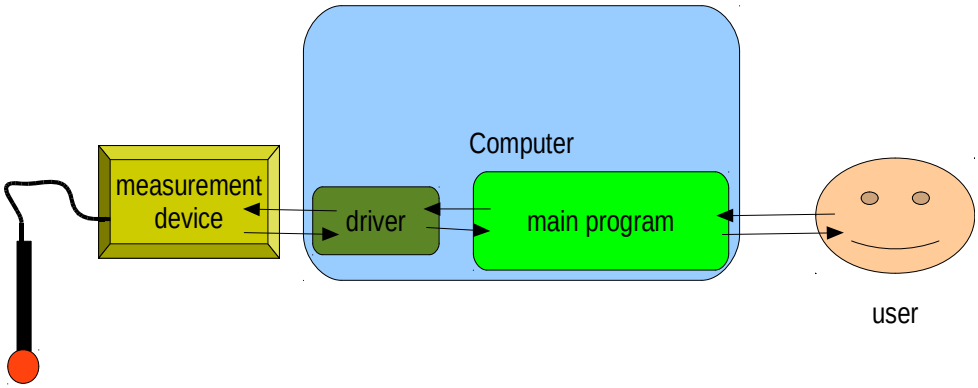


Fig. 1: Mensurasoft System

A professional scientist needs several scientific devices. For example, in ecology, we must measure accurately temperature AND pH AND light: there are not programs from devices makers for all them. A teacher of sciences needs to show to the students how to make an experiment. If there are several models of measurement devices in the classroom, with different software, the explanations will be very difficult for the teacher, who dreams about one piece of software for all these devices.

Drivers for measurement devices must be simple and easy to use, because science workers may not have great skills in programming. They must be usable with a lot of programming languages, in order to be used by many people.

Dynamic Libraries as Drivers

Dynamic libraries are pieces of compiled program, with functions callable by others programs; they are standardized and available in major operating systems: MS-Windows, Linux (named "shared objects") and Mac-OS [1]. Many programming languages can make dynamic libraries, often available as free software, but they must be compiled ones, such as: C/C++, Pascal, and some kinds of Basic.

On the other hand, almost all programming languages can use dynamic libraries, including office software such as Openoffice/LibreOffice, or the calculation specialists for example FreeMat [2], Scilab [3] Julia [4], R [5], and tools for general programing as Basic, C/C++, Pascal, Python, Ruby, Rebol.

In order to use dynamic libraries for all scientific devices, we must standardize the functions, giving them the same name in all the drivers. "Mensurasoft system" is a proposal for a possible standard: the names are short and consist of combination of letters.

Fundamental functions and additional properties

Computer must receive measures from the measurement device, and the driver needs a function for this input. In Mensurasoft system, the name of this input function has the letter "e" (as "entrée" in French, "eingang" in German, "entrada" in Spanish, "entrata" in Italian and "entry" in English). This is the main job.

If the device has an actuator, computer must send a value to the device, in order to change its state. In Mensurasoft system, the name of this output function has the letter "s" (as "sortie" in

French, "salida" in Spanish).

Inputs and outputs can be digital or binary if there are only two values as "yes/no", "1/0", "on/off", represented with the letter "b" in Mensurasoft. They are analog if there are a great number of values, for example when we measure temperature, voltage or pH: in Mensurasoft, the letter is "a" for them.

Computers use different type of numbers: integers or reals ("floating"). Reals are often coded as "double" (double precision), using 8 bytes (64 bits), and integers can be coded on 4 bytes (32 bits). When the result of an input function (or the value to be set by an output function) is an integer, there is no special letter, but when the result or the value is a "double", a letter "d" is added.

Sometimes (often) there are several inputs. For example, the little Arduino Uno has 6 analog inputs; a simple thermometer only measures temperature at one sensor, but we could have the results either in kelvins or in Celsius or even in Fahrenheit system, thus three (3) analog inputs are needed. So the input functions have one parameter meaning the channel for measurement which is usually coded as integer without a special letter; 0 corresponds to the first channel, 1 to the second and so on.

Output functions have two parameters. The first is the number of the channel, as for input functions, and the second is the value to be set by the device (integer or "double").

In programming languages, there are several ways to pass parameters to a function: stdcall, cdecl, pascal, and others. "stdcall" is more used in Windows world, "cdecl" in Linux world, but this is no mandatory. In order to use all of them, we must put a prefix "std" if the declared function is "stdcall" and "c" if the function is "cdecl".

For example, in Mensurasoft system, the first analog input channel sending a real (double) is stdead(0) with the calling convention "stdcall"; the second binary output is stdsb(1,1) if it is set to "On" and stdsb(1,0) if it is set to "Off"; with "cdecl" convention, the third binary input is ceb(2) and the fourth analog output set to 3.44 is csad(3, 3.44).

That is enough for main programming languages, but there are some among them which want other functions. Some languages accept parameters only as "double", even for the channel number: instead of "d", we put "double" to show that all parameters

and results are in double precision. Some languages deals with parameters and results only as "strings": we can add "str" at the end, and then the word stdeastr (0), for example, stands for the first analog input. Others need special functions (R, Scilab) in the dynamic library, but they are not included in the present work. A detailed outline of the naming procedure for functions in the Mensurasoft system is shown in Fig. 2.

Additional string functions

In Mensurasoft system each numerical function as already has been mentioned above has its name, coded by a string of characters. This string is a null-terminated string, as pchar in Pascal, zstring in FreeBasic, s in PureBasic or LPTSTR in C++. The name of the function begins by a "n" (as "name", "nom", "nomo", nome", "nombre"), e.g. nead(1) is the name of the second analog input. If a numerical function does not exist, its name is an empty string, i.e. a string whose length is 0; for example, cnead(133) is usually an empty string, because normal devices have only a few channels for measurement.

These "name functions" are very useful since:

a. when we do not know the channels of the device: we can ask cnead(0), cnead(1), cnead(2) and so on in order to find out them

b for general purpose software with menus and dialog boxes: to show all analog inputs the programmer is able to use a loop, asking the name of the analog input until this name is empty.

Additional string functions can be added for more comfortable use. A function "detail" could provide information about the device such as the company identity, the device name, the programmer's name, the date of programming etc. Even if there are not parameters, the name will be cdetail() and stddetail(). We could also define a title function, which will send a smaller string, for example only the name of the device, as: stdtitre() and ctitre(). This small string could be useful, e.g., as a title in a dialog box.

An optional function for calibration stdcalibration(ch) and ccalibration(ch) takes a string as input and gives a string as output. This function is useful only for specific devices, and the programmer can put anything in it.

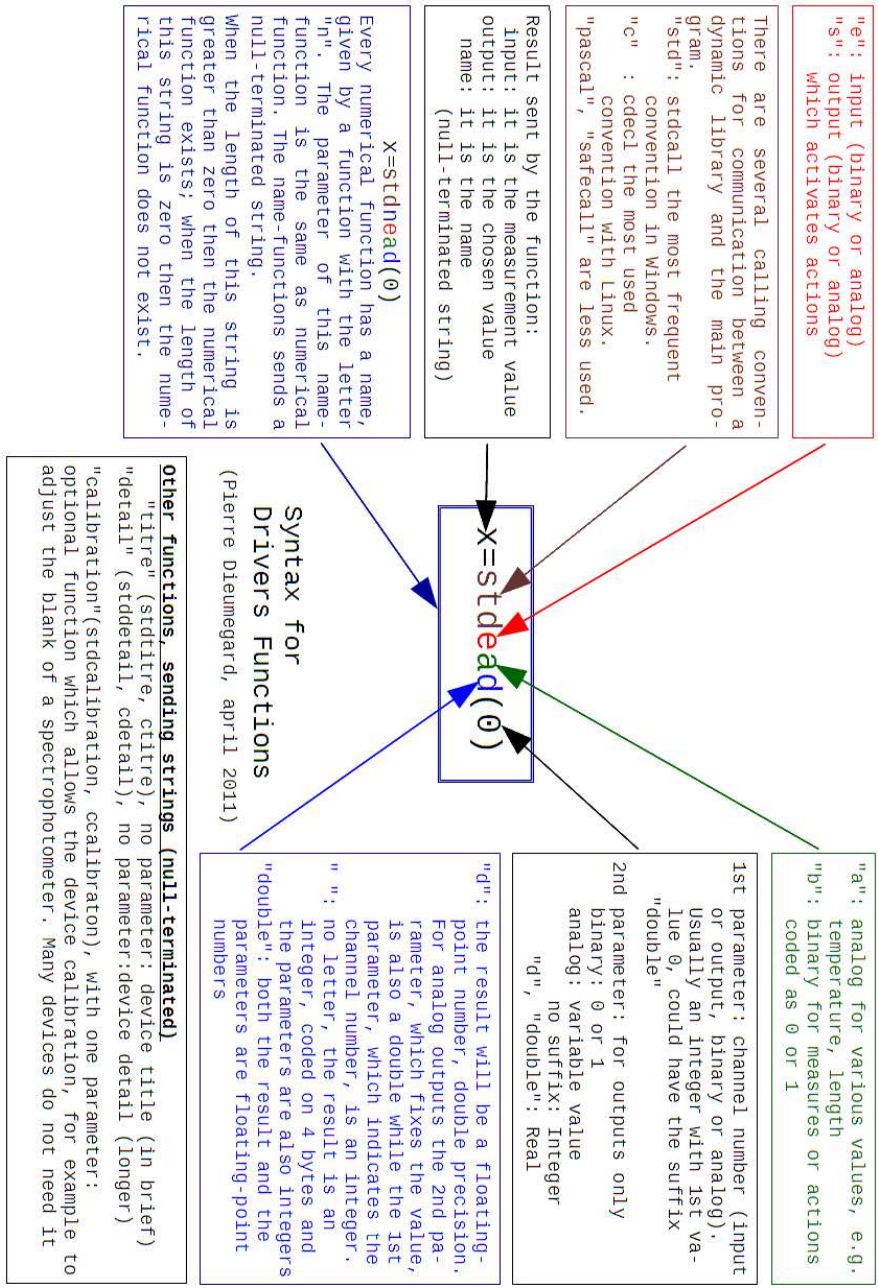


Fig. 2: Syntax of measurement function for Mensurasoft

A free project

Dynamic libraries are well known, and well standardized in programming languages. Mensurasoft is only a proposal for the adoption of a standard for the functions' names in order to allow all programmers to be able to use dynamic libraries of others.

Until now, all drivers and all application programs of Mensurasoft system are free software. The best license for drivers seems to be LGPL [6], because the drivers are small programs, and the best license for application programs seems to be GPL [7]. Nevertheless, we can imagine non-free drivers and non-free application programs, co-operating with free drivers and applications.

Three frequently asked questions need to be answered:

a. how to get existing drivers

b. how to make new drivers,

c. how to build programs calling them

a. The repository of the developed drivers for a lot of devices is at:

http://sciencexp.free.fr

mainly in French [8], but Google Translate may provide a decent translation for other languages. In this site, there are also application programs for Windows and Linux. They can use three (3) channels, for three (3) different drivers, make a graph of measured values, and save the values in files. They can transform the measured values by a formula (for example, to transform tension in volts from a pH-meter with analog output to pH). They are multilingual by language files.

Mensurasoft-PB [9] is written in PureBasic. A booklet in pdf is available in French, English, Spanish and Esperanto. Another program, Oscillo-PB, is very similar to Mensurasoft-PB, but is designed for fast measurement, like an oscilloscope. Fig. 3 shows a window from Mensurasoft-PB.

Mensurasoft-LZ is written in FreePascal/Lazarus. A booklet in pdf is available in French, English and Esperanto language. Fig. 4 shows an example of its usage in Linux illustrating the behavior of a common capacitor.

b. The best way to build a new driver is to modify an existing driver found on [8]. You can modify a "true driver" for a real device, or a "demonstration driver" [10].

Available drivers are in Delphi, FreePascal and Lazarus, FreeBasic, PureBasic, Oxygen-Basic, CodeBlocks, Dev-C++, C++Builder, Visual-C++.

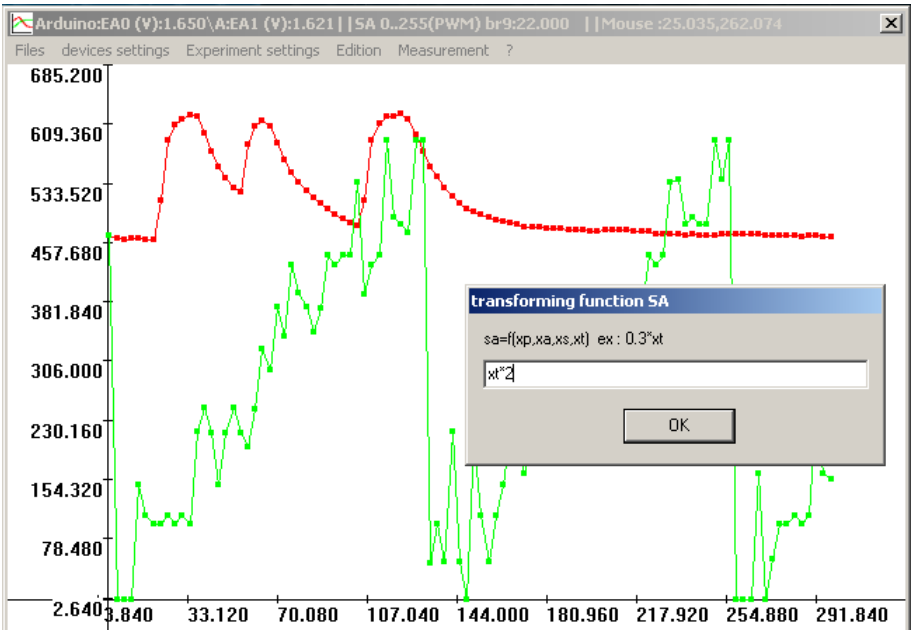


Fig. 3: An example window from Mensurasoft-PB

For example, if you have a new pH-meter, you can modify a driver only for a few functions: stdead, stdnead, stddetail, stdtitre. After compilation, you will be able to use Mensurasoft-PB, Mensurasoft-LZ and others with your new driver, and measure pH easily. Details of programming is available with booklets in French [11], English and Esperanto.

c. The booklets for programming also show how to use several languages to call these dynamic libraries: a lot of Basic, C/C++, Pascal, Logo, Python, Rebol, Ruby, Freemat, Scilab, Julia, R ...

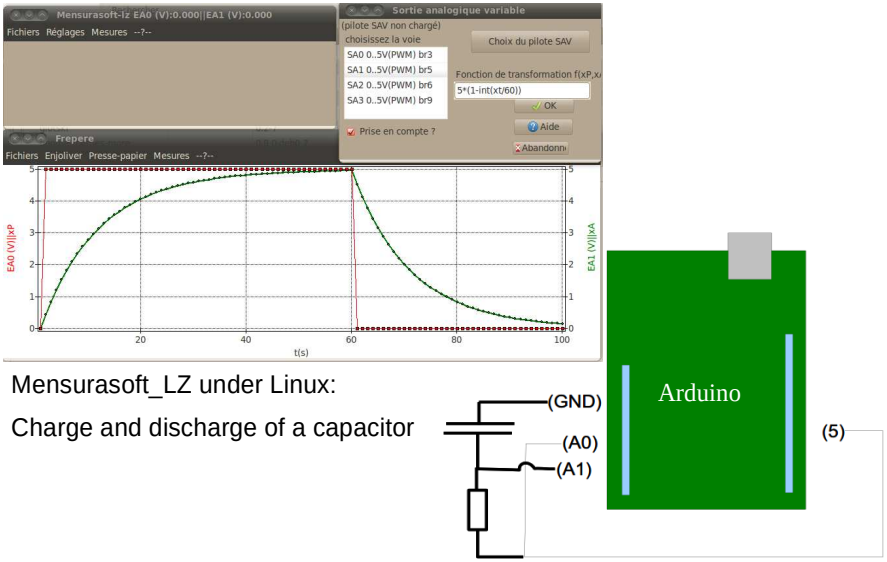
Small programs could also be found at [12].

Conclusion

Mensurasoft system evolved for twenty years, and works well, at least for slow measurement, and for Linux and MS-Windows operating systems. For fast measurement, like an oscilloscope, limitations are due to the speed transfer of USB. A rather simple but systematic notation of functions implemented in dynamic libraries has been adopted and successfully applied and tested. The same system is available for Mac OS, but never

has been tested.

The author encourage those who would like to be involved to this continuing effort and to contribute their driver for another device to the public domain via a link on the website [8] or to share a new written related application program. Do not hesitate to contact the author for any comment or related work.



Mensurasoft_LZ under Linux:

Charge and discharge of a capacitor

Fig. 4: Application of Mensurasoft-LZ in Physics

References

[1] Dynamic-link library: https://en.wikipedia.org/wiki/Dynamic-link_library
[2] FreeMat: http://freemat.sourceforge.net/
[3] Scilab: Scilab Enterprises, http://www.scilab.org/
[4] Julia: http://julialang.org/
[5] R: http://www.stats4stem.org/r-math-functions.html
[6] GNU Lesser General Public License: https://en.wikipedia.org/wiki/GNU_Lesser_General_Public_License
[7] GNU General Public License: https://en.wikipedia.org/wiki/GNU_General_Public_License

[8] Sciencexp - the site of the Experimental Sciences: http://sciencexp.free.fr
[9] Mensurasoft-PB: http://sciencexp.free.fr/index.php?perma=Mensurasoft-PB
[10] Pilot demonstration: http://sciencexp.free.fr/index.php?perma=pilotes_demonstration
[11] Mensurasoft: drivers of measurement devices and actuators system under MS-Windows and Linux: http://sciencexp.free.fr/index.php?perma=Programmation
[12] Demonstration applications: http://sciencexp.free.fr/index.php?perma=Applications_demonstration

Previous Publication in FUNKTECHNIKPLUS # JOURNAL

None

*** About The Author**

Pierre Dieumegard, was born in Limoges, France, in 1955. He teaches biology, geology and computer science in Lycée Pothier in Orléans, France. Since his thesis in plant genetics (1984), he started to use computers. After that, for many years, he was involved in research on the use of computers in science education.

pierre.dieumegard@ac-orleans-tours.fr