# FluidLab : research on a modern, clean and open-source code for laboratory experiments

http://fluidlab.readthedocs.org

Pierre Augier

LEGI, CNRS, Université Grenoble Alpes

12 October 2015

## Software to control computers to do lab experiments

- control of devices (need drivers)
- complex objects (devices working together)
- classes of devices and objects
- close loop (PID)
- temporal loops
- tasks to do in parallel, “at the same time”
- many experiments to span parameters
- save data in readable formats, load and plot it
- easily modified
- interactive, scripts and GUI
  (GUI for the exp $\neq$ GUI to develop)

## Diverse possibilities

**Labview (proprietary software by National Instruments)**

Pros

- very adapted to "National Instruments"
- pretty good documentation
- somehow easy, in particular for simple GUI
- graphical programming

Cons

- difficult for many researchers
- graphical programming
- close source (black box)
- cost (on the long range)
- very specialized language (analyses and plots have to be done with another tool)

## Diverse possibilities

**Digiflow (house-made proprietary software)**

Pros

- good for images
- scripting language
- post-processing

Cons

- close-source,
- bad buggy scripting language
- no tests (?)
- not portable (only Windows)

## Diverse possibilities

**Old-school house-made software, often in low level language (C)**

Pros

- free (like "free beer")
- sometimes open-source

Cons

- difficult
- often quick and dirty
- often messy
- no documentation
- no tests
- not portable

## Types of instruments, connections and communications

**Connections and communication**
GPIB, serial (RS232, RS485), USB, Ethernet-RJ45, Firewire...
VISA library (abstraction)

**Types of control**

- with a signal (for example for motor, camera trigger)
- with strings (with norms, for example derived from "GPIB language" + classes of device: IVI, IEC60488)
- with normalized protocols (Modbus, CANopen, etc...)
- with libraries (often closed): DAQ-NI, Comedi, Firewire, ...

## Other approach: clean open-source

FluidLab (part of the FluidDyn project)

Use tools and methods of modern programming:

- Python (high-level generalist dynamic language)
- object-oriented
- distributed revision control tools (Mercurial and Bitbucket)
- very easy installation
- semi-automatic documentation
- unit tests

**In building !**

Still only alpha versions

## Example 1: control a power supply

Own "language" with serial through USB

- documentation
- interactively and in scripts
- temporal loop

`http://fluidlab.readthedocs.org/en/latest/examples/loop_with_1instr.html`

- internal: simple + automatic documentation

## Example 2: control a motor and its frequency drive

### Modbus RTU with RS485 in RJ45!

`http://fluidlab.readthedocs.org/en/latest/examples/control_motor.html`

- driver for the motor
- documentation
- interactively + scripts + GUI
- internal: quite simple + automatic documentation

## Span parameters and auto-organize the data

Most of the time, one experimental apparatus is used for many experiments with different input parameters.

It is very convenient if the results can be organized automatically in directories.

**Experimental session**

```
from fluidlab.exp import Session
session = Session(path='Tests', name='False_exp')
```

http://fluidlab.readthedocs.org/en/latest/examples/session_instru.html

- log information
- send emails
- save and organize data
- plot figures
- can be reloaded

## Conclusions

**FluidDyn**: a project for open-science in fluid dynamics with Python

**FluidLab**: an attempt for a modern, clean and open-source code for experiments

- can already do many things
- in building: will do many more!

**Future**: users, developers, community ?

We need a model for open-software in academics!

Still to be invented!