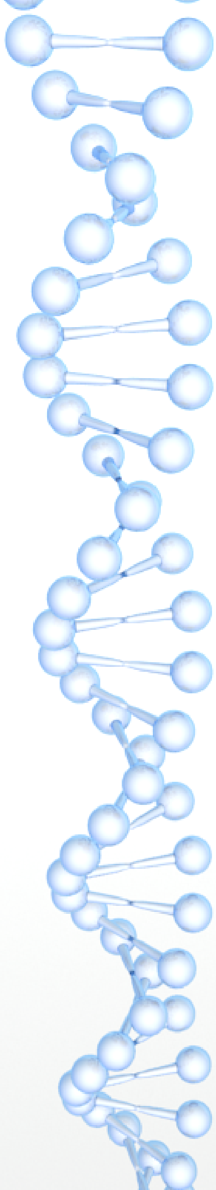
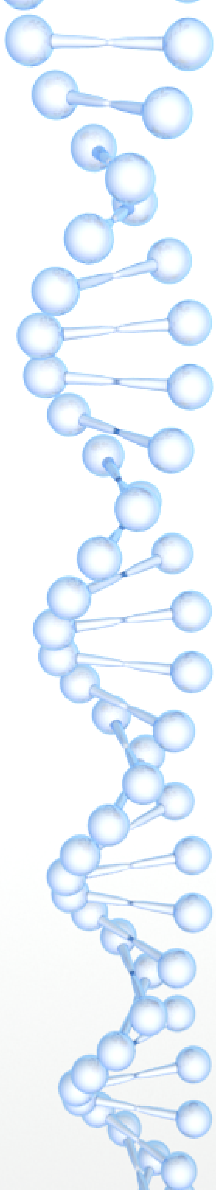


Present and Future of the IPOL Journal

Machine Learning Applications

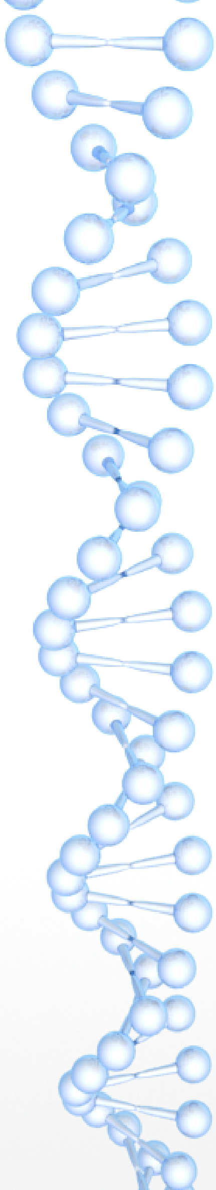
Miguel Colom
http://mcolom.info

CMLA, ENS Paris-Saclay



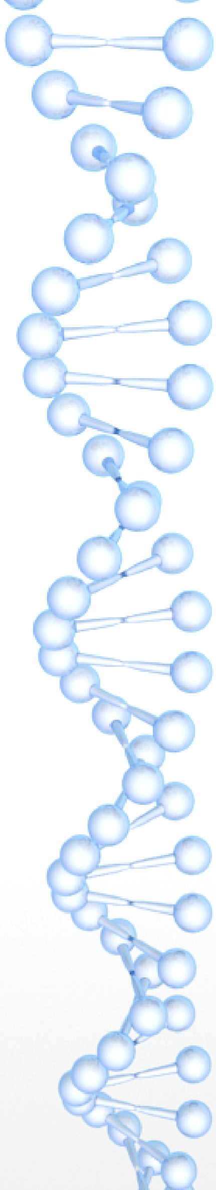
What is Reproducible Research?

- It redefines the **result of the research**.
- It is **not just a paper**!
- It is:
  - The research **article**
  - The **computational facility** which recreates the results given the input data
  - The **source code**
  - The **data**



Why Reproducible Research is Needed?

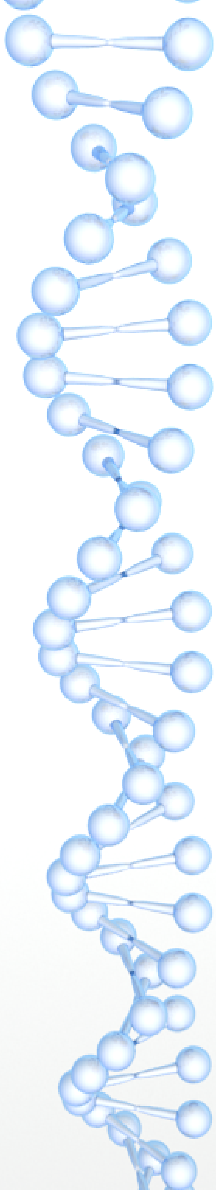
- The **results** in the published paper may be **wrong**
- The results in the published paper may not be **generalizable** (only work with that particular images)
- The procedure described in the paper may be **inaccurate**, **vague**, or **incomplete**.
- Published paper usually **don't have space enough to describe all details** (which are need to implement a method!). For example: all parameters in algo subalgorithm. Pre/post processing steps?
- The **source code** of the method **may not be available** or the author may refuse to make it public (because of a patent, for example)
- A paper doesn't allow to **compare** results with other methods



Reproducibility and Repeatibility

- **Reproducibility**: the ability to obtain the same results of a reference experiment.
- **Repeatibility**: the ability to perform the experiment as many time as needed.

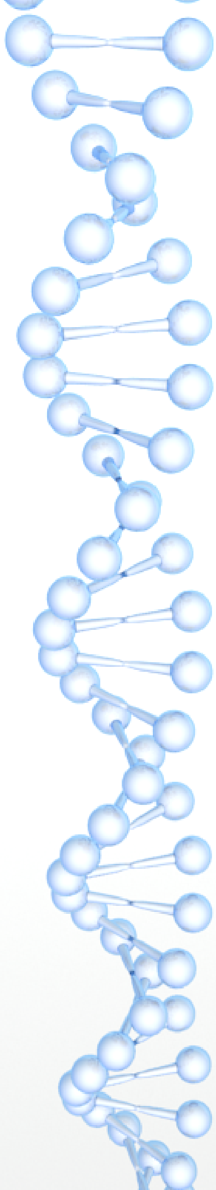
- Both are needed in Reproducible Research.
- Always easy to obtain? **No**! For example, in Biology:
  - The experiment may not be reproducible since it is impossible to know all the exact conditions
  - Repeating a biological experiment may take months or even years! (for example, studying the effect of a gene mutation in mice generations).
- **Do not trust** the authors. Just **try** and **evaluate** the algorithm **yourself**.



Reproducibility and Repeatability in Image Processing

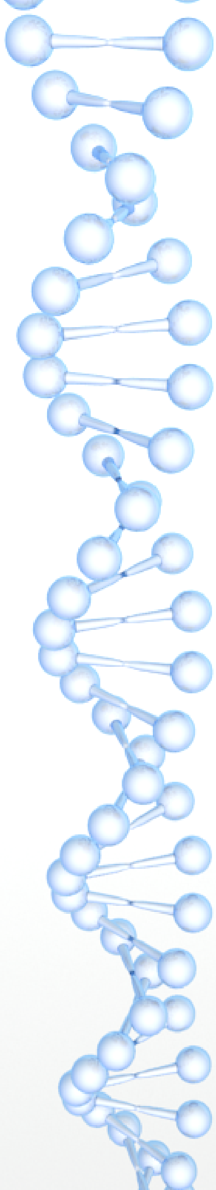
- **Similar to Biology**? **NO**!

- Since the algorithms are completely **well defined** using proper mathematical descriptions, it is possible to **reproduce** and **repeat** all experiments.



Reproducibility and Repeatibility in Image Processing

- Then, **why not always done**?
- Several reasons:
  - The **source code** of the authors is **not of enough quality** (readability, correctness, usability, style) and **they don't want to make it public**.
  - The source code **might not do exactly what is described** in the paper.
  - It **takes more time** to properly design a good demonstrator than writing a paper
  - Not only the **paper** will be peer-reviewed, but also the source code of the **algorithm** (checking that is does exactly what is described in the paper).

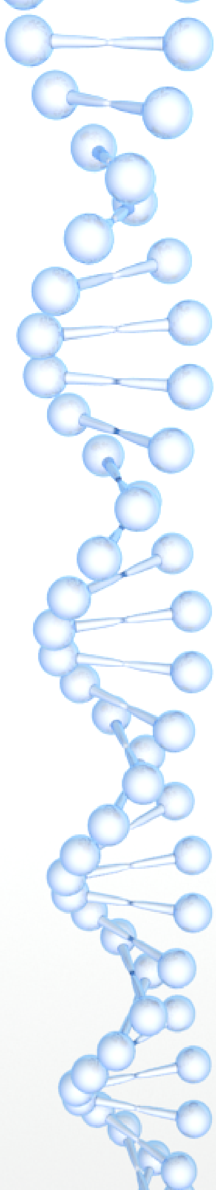


Summary

- **software** is part of the **research work**

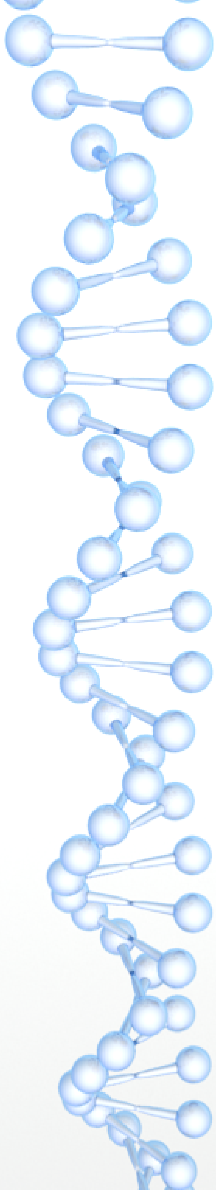
- software needs be **published**
  - with a **review process**
  - with **quality criteria**

- In the particular case of IPOL we have:
  - **detailed algorithms**
  - **verified** and **usable code**
  - instant test **demos**



Benefits of having a RR demonstrator

- **Comparisons** with other methods **easier**
- The results can be **trusted**, since they passed an exigent **peer-review** process
- The number of **citations** of the article grow when the scientific community can **reproduce** and **compare** the results of the methods (look at IPOL articles)
- More **visibility** to the paper
- **Convincing** results

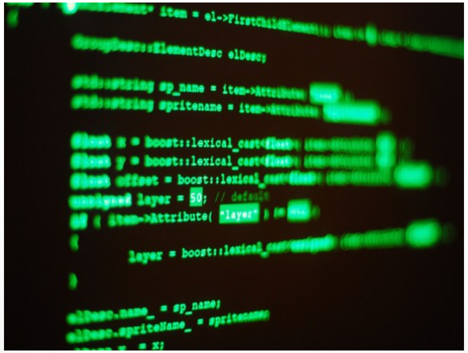


Disadvantages for the author

- It takes **much more time** than writing a regular article in a journal
- The submitted software must comply with some **strict software guidelines**: readability, documented, portable, standard
- The **peer-review** process is in general more **exigent** and thorough compared to a ordinary journal

What is IPOL?

- An Image Processing (OnLine) **journal**
- Each article: **source code + PDF article + demo**
- All **peer reviewed**
- **Open source** software

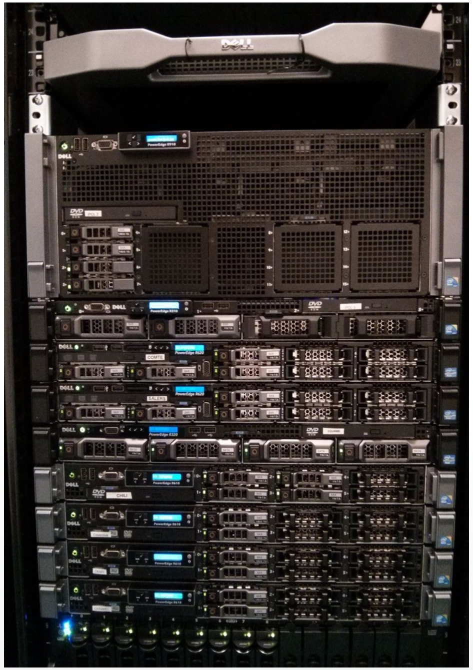


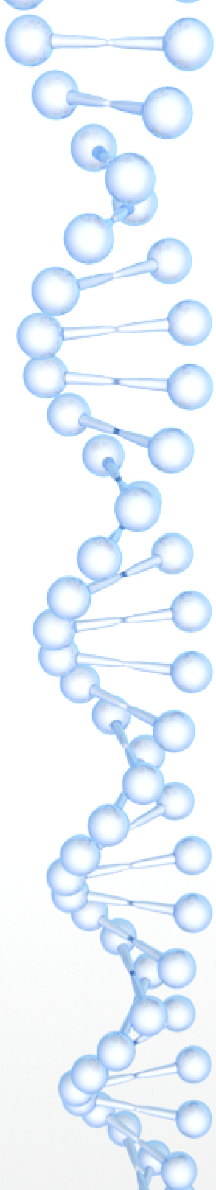
```
[illegible]* item = el->FirstChildElement( [illegible]
GroupDesc::ElementDesc elDesc;

std::string sp_name = item->Attribute( [illegible]
std::string spritename = item->Attribute( [illegible]

float x = boost::lexical_cast<float>( [illegible]
float y = boost::lexical_cast<float>( [illegible]
float offset = boost::lexical_cast<float>( [illegible]
unsigned layer = 50; // default
if ( item->Attribute( "layer" ) != [illegible]
{
    layer = boost::lexical_cast<unsigned>( [illegible]
}

elDesc.name_ = sp_name;
elDesc.spriteName_ = spritename;
elDesc.x_ = x;
```





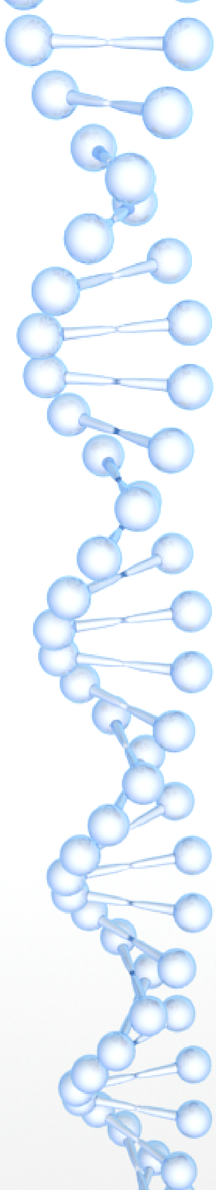
IPOL demos workflow

1) Users **choose** an **input** (or **upload** their own)

2) The user **sets the parameters** of that image and click on the “**Run**“ button.

3) The **algorithm** (C/C++, Matlab, …) is **executed** and the **results** are **shown** in the browser.

4) The results are **archived** if the input image was uploaded.

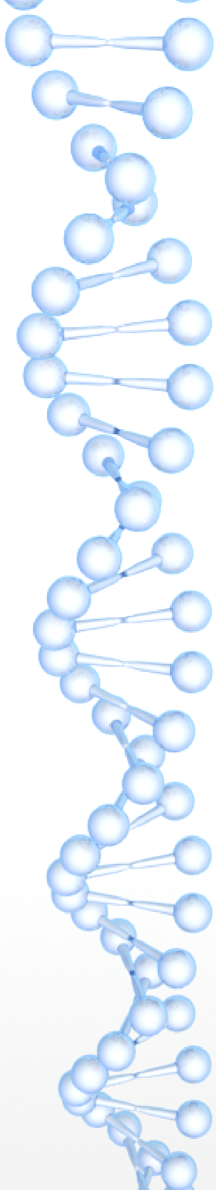


IPOL

Let us see it in action!

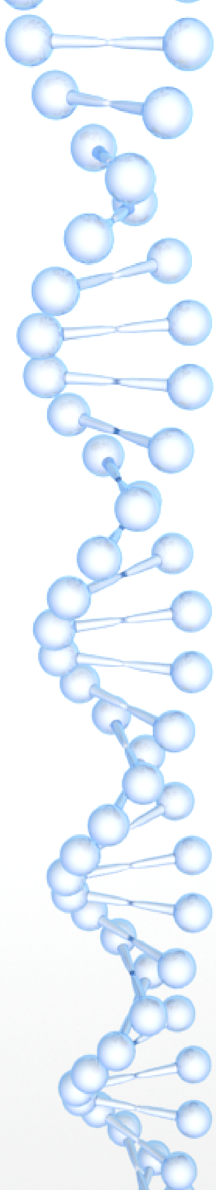
Criticism to IPOL

- Excessive **efford** to arrive at a **reproducible article**
- Length and **duration** of the **peer-review** reports --> We’ve started a new **fast review process** in June 2018.
- Lack of an official **Impact Factor**
- Lack of **GPU** --> They’ll be available in **2018**



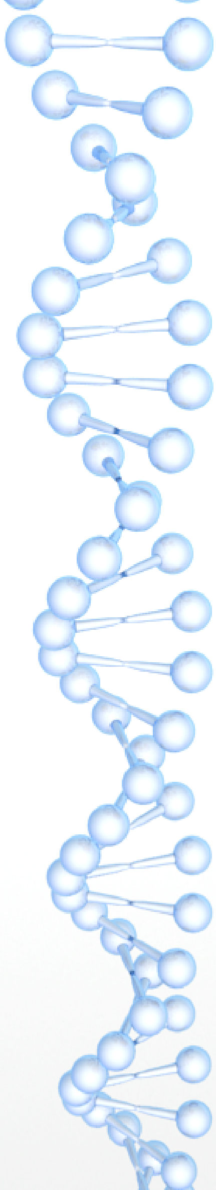
The present (I)

- IPOL: more than **8 years publishing**
- Well-established journal
- **133** articles **published**, **4** articles **accepted**, **14 preprints**
- **261.197 unique visits** in 2017. **Code/data** downloaded **12.173 times**
- **Archive**: more than **250,000** experiments in 8 years
- More tan **1,000,000 online executions** in 8 years



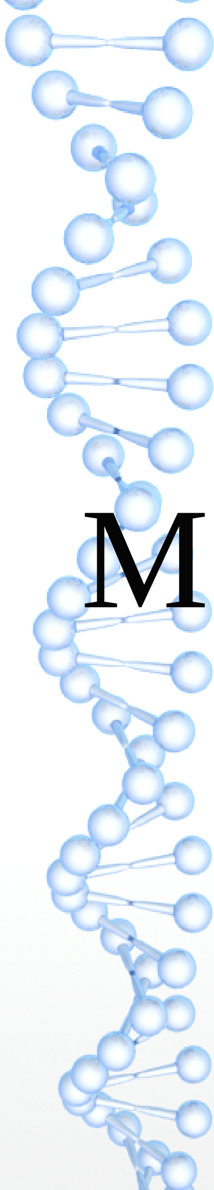
The present (II)

- **MATLAB** and **Python** also **accepted languages**
- Servers with **GPU** to arrive
- The most relevant **topics in Image Processing** are already in IPOL (classic and state of the art)
- Demo system: **full architecture of microservices.**
- **Video demos** are possible
- Indexed by **SCOPUS**. In the Thomson-Reuters ***Emerging Sources Citation Index***



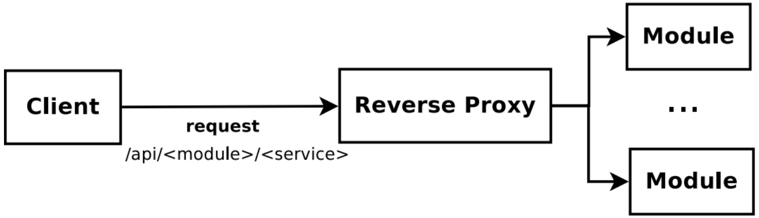
The future

- Obtain an **Impact Factor**
- Improve the overall **design** of the **website**
- **Extend IPOL** from only Image/Video Processing to **Machine Learning Applications**



Extension to
Machine Learning Applications

Current Architecture



Client

request

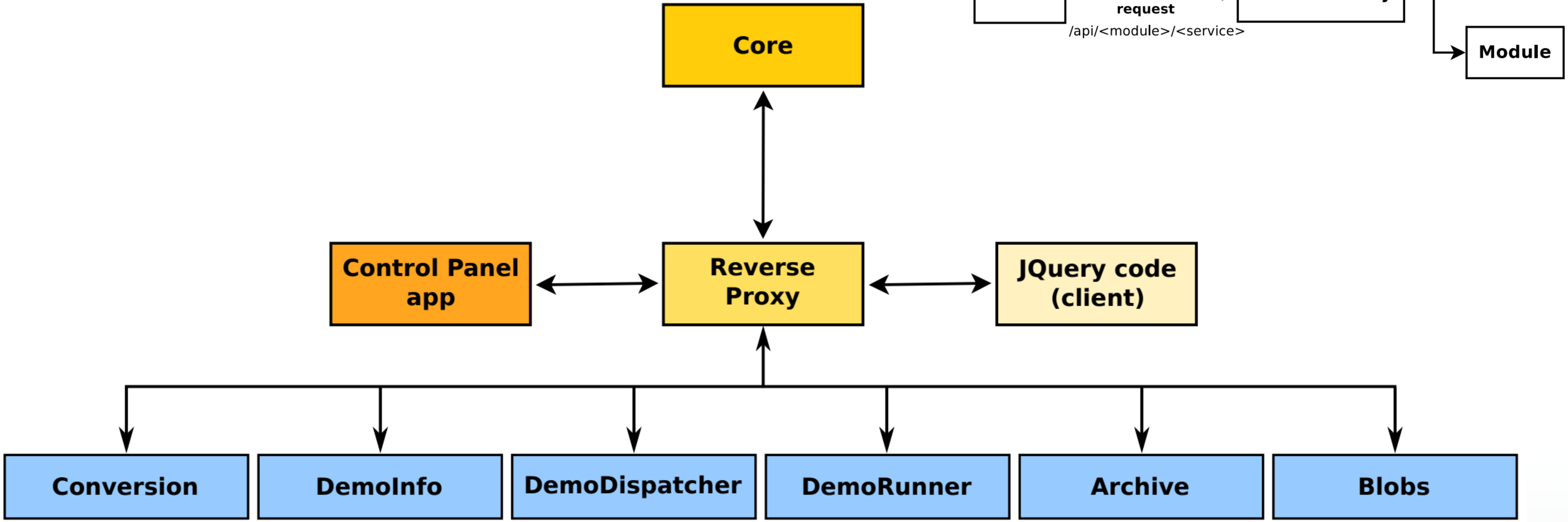
/api/<module>/<service>

Reverse Proxy

Module

...

Module



Core

Control Panel app

Reverse Proxy

JQuery code (client)

Conversion

DemoInfo

DemoDispatcher

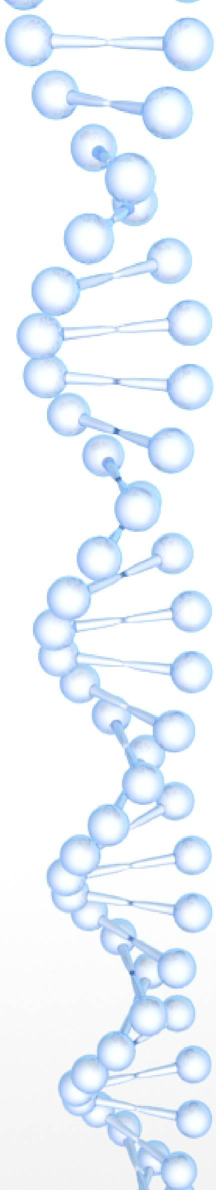
DemoRunner

Archive

Blobs

Limitations

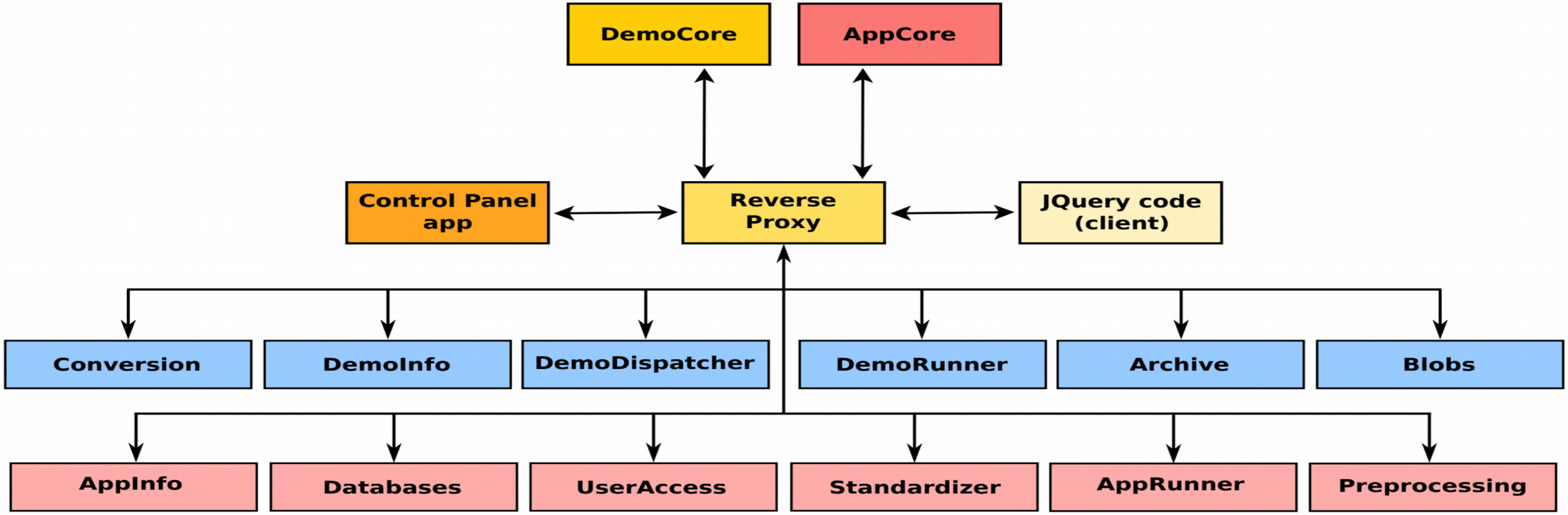
- OK for **isolated demos**. Can add new datatypes (video, audio, 3D, interactive controls, ...)
- Demo: **standalone**. **Does not share information** with the others. **Stateless**.
- **Not well adapted** to **Machine Learning applications**
- New concept: **Application**
- **Major changes** in the **architecture** needed



Demo vs Application

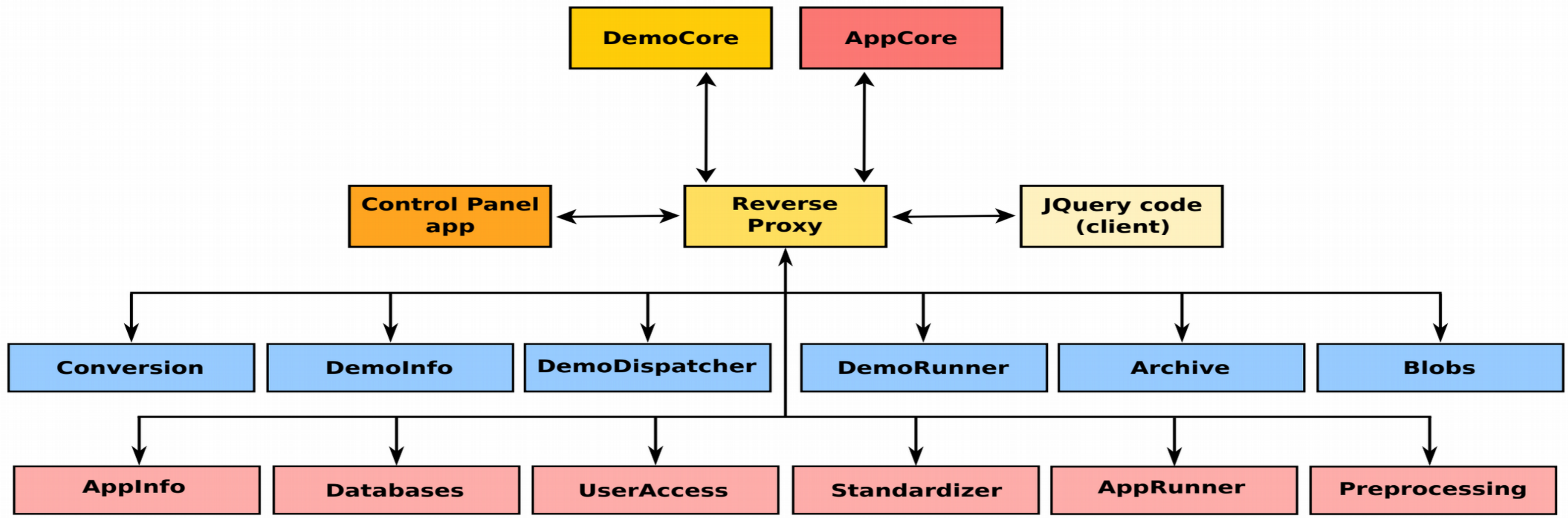
- **Execution time**: demo start and **ends** shortly. Application does **never ends**.
- The Application can be ***sleeping*** when there’s no activity. It might **wake up** when a new experiment is added to the archive, to learn. Can wake up regularly to perform needed tasks.
- **ML** applications are **more complex**:
  - **Pre-processing more complex (**more **heterogeous, less structured)**
  - **Standardization** of the data
  - **Access permission** to the data by different types of users

New ML Architecture



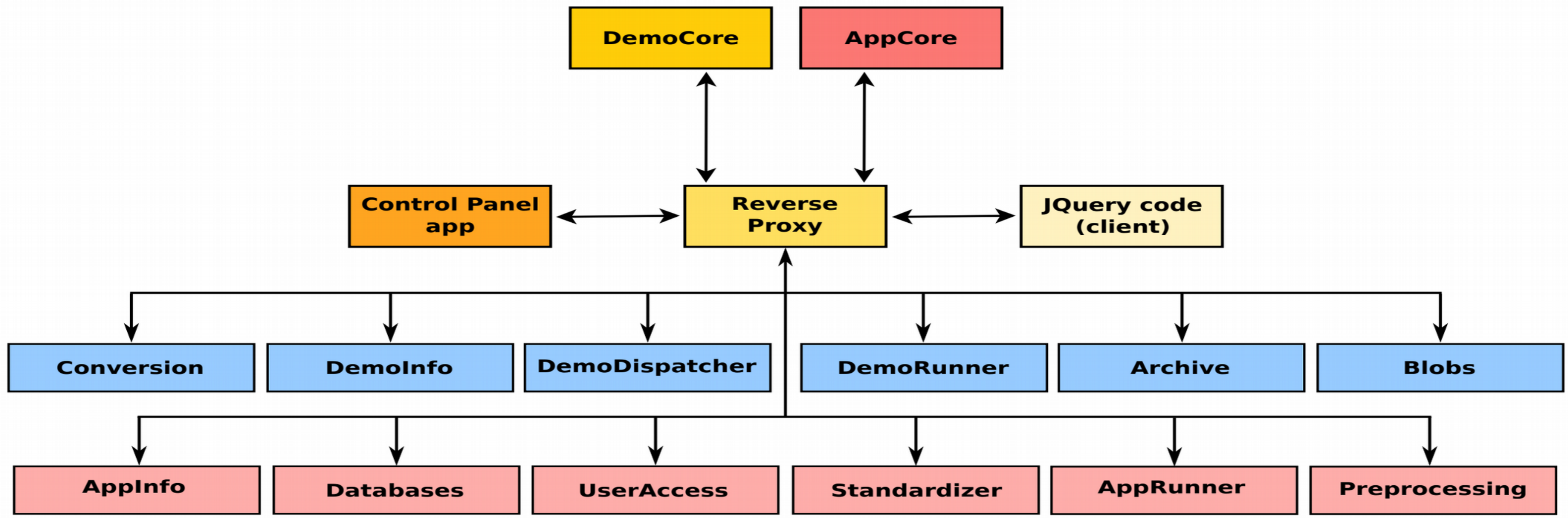
AppCore: same rol as current DemoCore, but to control the execution of Applications

New ML Architecture



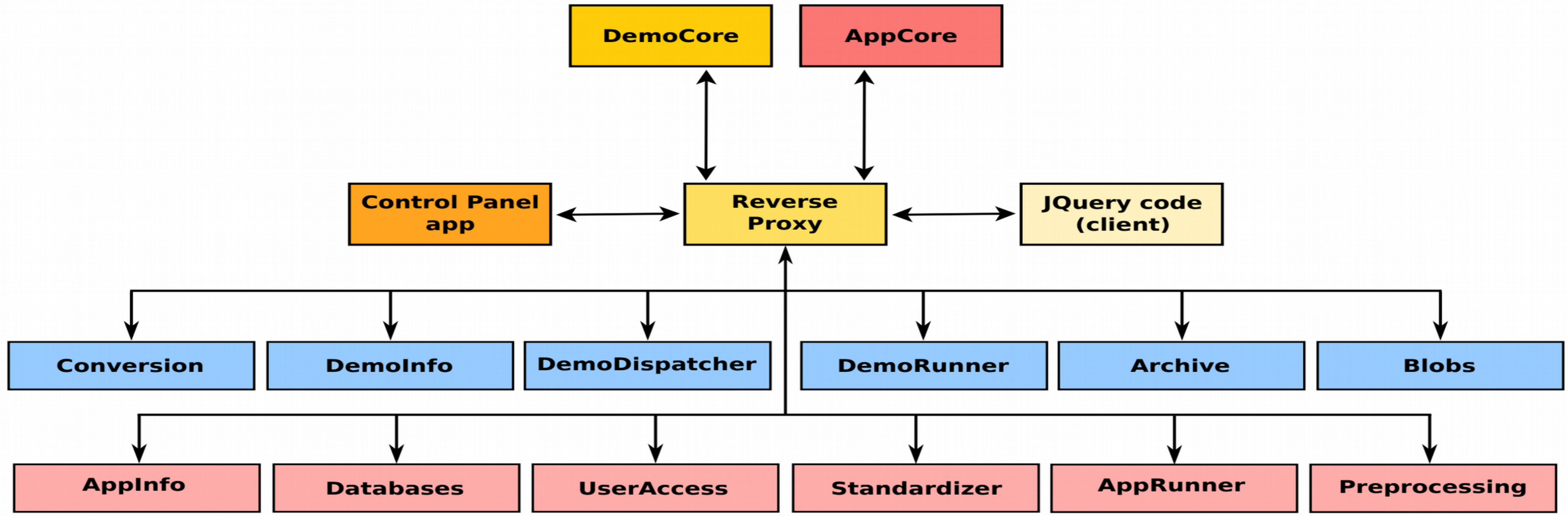
Databases: storage and management of training and testing datasets

New ML Architecture



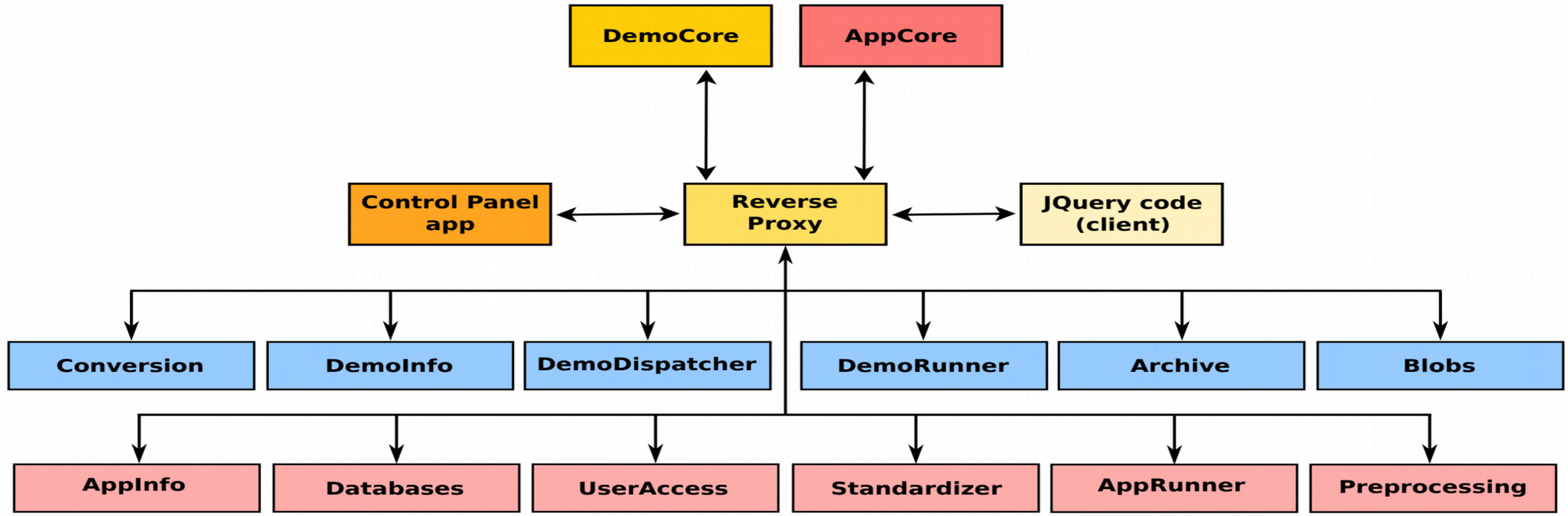
UserAccess: lists of users and authorization management

New ML Architecture



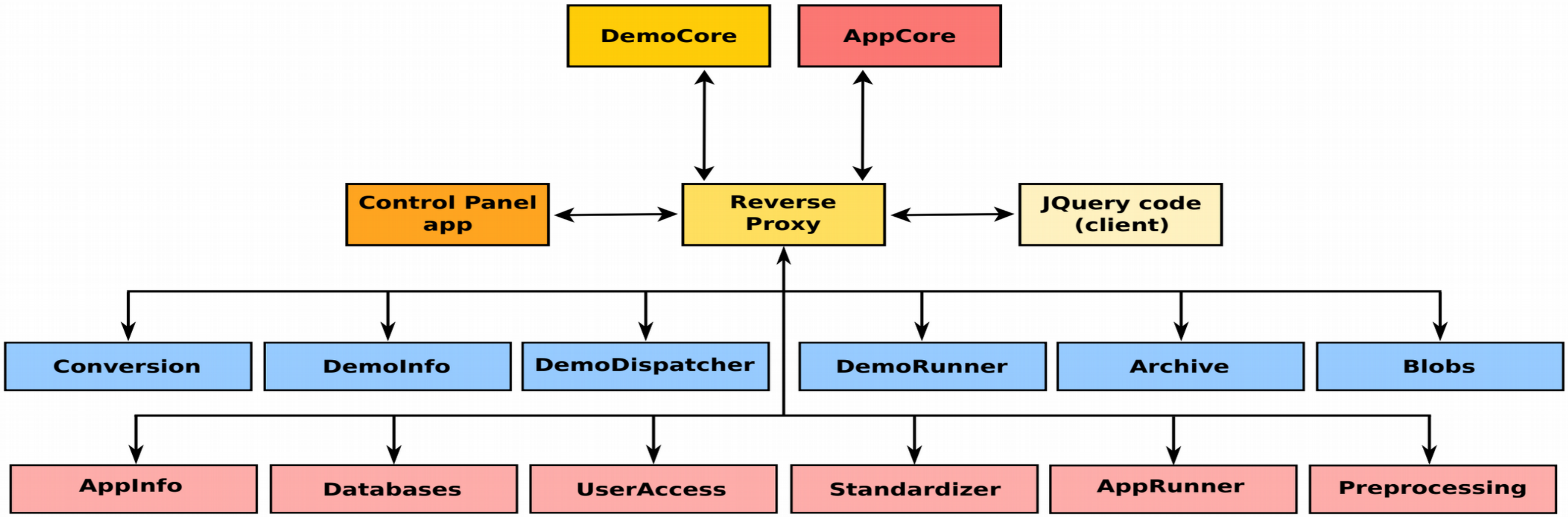
Standardizer: to structure the data in a format understood by the system and the algorithms

New ML Architecture



Preprocessing: treatment of heterogeneous data in order to process it by the algorithm. For example: treat NAs, normalize variance, change sampling rate of sensor, etc.

New ML Architecture



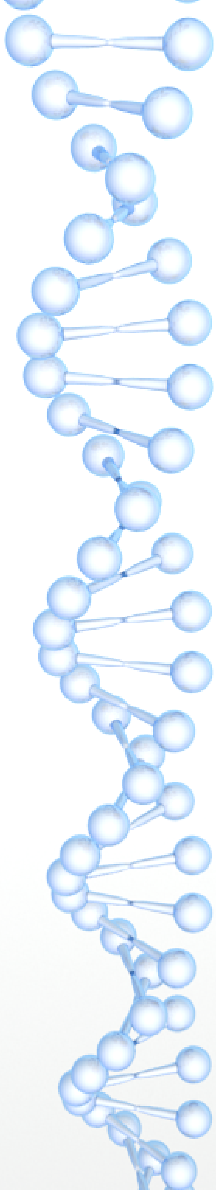
Archive: allow structured access of a demo to its own and eventually other demos’s experiments (according to the configured permissions).

Wanted features: **comparison of algorithms**

- Need to define **formats** to **homogenize** the outputs of diverse algorithms. The **Standardizer** module will do it.
- In the case of missing data or if **preprocessing** is needed, then the **Preprocessing** module will be invoked too.
- For **evaluation**: datasets are previously **annotated** by experts and serve as a **ground-truth** to evaluate the algorithms. The module responsible for storing the testing and learning databases: **Databases**

Wanted features: **chaining of algorithms**

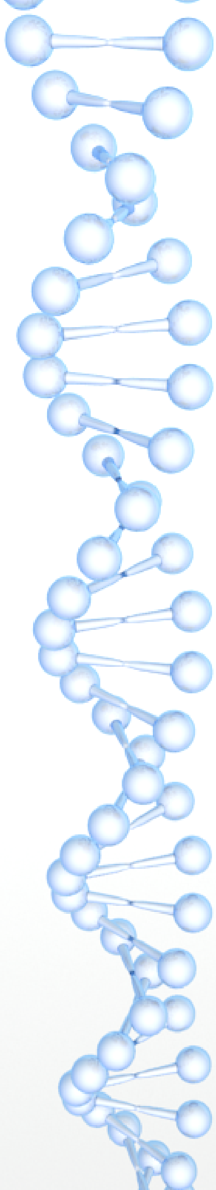
- In short: **connect** the **output** of one algorithm with the **input** of the other.
- The **Standardizer** module will take care of choosing the right format
- **No need** to manually **write wrappers**



Conclusions

- IPOL managed to create a **simple system** for **editors/users** to create demos quickly
- **Archive** of more than **250,000 experiments.**
- More than **1,000,000 executions** so far
- The journal is **used** by **many academics** and **industrials**
- **Not an Impact Factor yet**. This **discourages authors** to submit their work
- We expect a **large impact** in the **ML community** with the **extended system**

- **Free to use. Free software. Free knowledge. Please contribute! :)**



Thank you for your attention!

Miguel Colom
http://mcolom.info

CMLA, ENS Paris-Saclay