Open-Source Software Maintenance: Experience from IPOL

Pascal Monasse <pascal.monasse@enpc.fr>
LIGM, École des Ponts, Univ. Gustave Eiffel, Marne-la-Vallée, France

Sous la co-tutelle de :
CNRS
ÉCOLE DES PONTS PARISTECH
UNIVERSITÉ GUSTAVE EIFFEL

Workshop Open Science, Bilbao, Nov. 10-12, 2022

Software as Integral Part of Research Work

Software needs to be published

- with a review process
- with quality criteria

In the case of IPOL, we have

- detailed algorithms
- verified and usable code
- instant test demos

IPOL Software Guidelines

Requirements

- All software should have an open source license.
- The original authors may not be available to maintain the software.
- The IPOL team should be able to deal with minor adaptations

IPOL Software Guidelines

Requirements

- All software should have an open source license.
- The original authors may not be available to maintain the software.
- The IPOL team should be able to deal with minor adaptations

Consequences

- Originally, only ISO C/C++ code was accepted.
- Limited libraries with a stable API were authorized: Eigen (linear algebra), GSL (GNU Scientific Libraries), libpng, libjpeg, libtiff (image i/o).
- There was some "glue" Python code for the demo system.
- Reviewers should make sure the source is commented and does not use overly complex optimizations (keep it simple).

IPOL Meets Reality

Stringent requirements limited the submissions

- Additional libraries are allowed, should be embedded in the source archive.
- Matlab very popular in signal/image processing, but no standard, evolving API.
  $\rightarrow$ Encourage GNU Octave compatibility.
- Python with NumPy, same problems.
  $\rightarrow$ Minimize dependencies by using virtual environments with PIP (file `requirements.txt`).

Library Development for RR

Contribution to RR from library development

- Way to diffuse research results to larger use.
- Documented and referenced algorithms easy to be used from new PhD student.
- Extend the classic academic results to real applications.
- Gather algorithms and make easier comparisons and use in other context.
- Make easier software demonstrator or online demonstrations.

Library Development for RR

Contribution to RR from library development

- Way to diffuse research results to larger use.
- Documented and referenced algorithms easy to be used from new PhD student.
- Extend the classic academic results to real applications.
- Gather algorithms and make easier comparisons and use in other context.
- Make easier software demonstrator or online demonstrations.

Attention key point towards reproducibility

- Compatibility problems: libraries evolve rapidly (compilation issues).
- Implementation change: can change numerical results from a version to another.
- Implies the use of a container solution like *Docker* to ensure longer terms reproducibility.

Example of libraries of vision/imagine/geometry domains

Example of library of vision/image/geometry domains

Library	domain	langage	version	#auth.	date	funding
OpenCV	Comp. Vision	C++	4.5.5	1,383	1999	Willow Garage
ITK	Image Processing	C++/Pyt.	5.2.1	265	2000	Kitware
PCL	Point clouds	C++	1.12.1	464	2010	Willow Garage
CGal	Geometry proc.	C++	5.4	123	1996	Acad./GeometryFactory
CImg	Image processing	C++	3.1.2	72	1999	Acad.
Geogram	Geometric algorith.	C++	1.7.8	7	1998	Acad./INRIA/ERC
Olena	Image processing	C++/Pyt.	2.1	50	2001	Acad. / Project
Tulip	huge graph visualiz	C++/Pyt.	5.6.2	9	2001	Acad./private
Vigra	Comp. Vision	C++	1.11	50	2008	Acad.
DGtal	Digital geometry	C++/Pyt.	1.2	27	2011	Acad. / Project
OpenMVG	Mult. View Geom.	C++	2.0	86	2013	Acad./Mikros/Foxel
TTK	Topology ToolKit	C++	1.0	36	2017	Acad. / Project
Higra	Graph analysis	C++/Pyt.	0.6.5	4	2018	Acad. / Project

Example of libraries of vision/imagine/geometry domains

Example of library of vision/image/geometry domains

- Some libraries were initiated/funded by private company.

Library	domain	langage	version	#auth.	date	funding
OpenCV	Comp. Vision	C++	4.5.5	1,383	1999	Willow Garage
ITK	Image Processing	C++/Pyt.	5.2.1	265	2000	Kitware
PCL	Point clouds	C++	1.12.1	464	2010	Willow Garage
CGal	Geometry proc.	C++	5.4	123	1996	Acad./GeometryFactory
CImg	Image processing	C++	3.1.2	72	1999	Acad.
Geogram	Geometric algorith.	C++	1.7.8	7	1998	Acad./INRIA/ERC
Olena	Image processing	C++/Pyt.	2.1	50	2001	Acad. / Project
Tulip	huge graph visualiz	C++/Pyt.	5.6.2	9	2001	Acad./private
Vigra	Comp. Vision	C++	1.11	50	2008	Acad.
DGtal	Digital geometry	C++/Pyt.	1.2	27	2011	Acad. / Project
OpenMVG	Mult. View Geom.	C++	2.0	86	2013	Acad./Mikros/Foxel
TTK	Topology ToolKit	C++	1.0	36	2017	Acad. / Project
Higra	Graph analysis	C++/Pyt.	0.6.5	4	2018	Acad. / Project

Example of libraries of vision/imagine/geometry domains

Example of library of vision/image/geometry domains

- Some libraries were initiated/funded by private company.
- The others come mainly from university initiatives.

Library	domain	langage	version	#auth.	date	funding
OpenCV	Comp. Vision	C++	4.5.5	1,383	1999	Willow Garage
ITK	Image Processing	C++/Pyt.	5.2.1	265	2000	Kitware
PCL	Point clouds	C++	1.12.1	464	2010	Willow Garage
CGal	Geometry proc.	C++	5.4	123	1996	Acad./GeometryFactory
CImg	Image processing	C++	3.1.2	72	1999	Acad.
Geogram	Geometric algorith.	C++	1.7.8	7	1998	Acad./INRIA/ERC
Olena	Image processing	C++/Pyt.	2.1	50	2001	Acad. / Project
Tulip	huge graph visualiz	C++/Pyt.	5.6.2	9	2001	Acad./private
Vigra	Comp. Vision	C++	1.11	50	2008	Acad.
DGtal	Digital geometry	C++/Pyt.	1.2	27	2011	Acad. / Project
OpenMVG	Mult. View Geom.	C++	2.0	86	2013	Acad./Mikros/Foxel
TTK	Topology ToolKit	C++	1.0	36	2017	Acad. / Project
Higra	Graph analysis	C++/Pyt.	0.6.5	4	2018	Acad. / Project

Library Development for RR

Elements to increase reproducibility across libraries

- Limit dependencies to other libraries or include them inside the library itself.
- Orient to an header only library (if C++ like CImg or CGal since version 5).
- From a user perspective, extract all dependent header files.

Elements to increase reproducibility across libraries

- Limit dependencies to other libraries or include them inside the library itself.
- Orient to an header only library (if C++ like CImg or CGal since version 5).
- From a user perspective, extract all dependent header files.

Examples in C++:

It is possible to use a special option in order to list all header files:

```
g++ hello.c -o hello -I/specialHeaderLibPath -MD
```

Then command will generate a `hello.d` file containing all header files that are really used by the compilator to produce the executable.

Elements to increase reproducibility across libraries

- Limit dependencies to other libraries or include them inside the library itself.
- Orient to an header only library (if C++ like CImg or CGal since version 5).
- From a user perspective, extract all dependent header files.

Examples in C++:

It is possible to use a special option in order to list all header files:
`g++ hello.c -o hello -I/specialHeaderLibPath -MD`

Then command will generate a `hello.d` file containing all header files that are really used by the compilator to produce the executable.

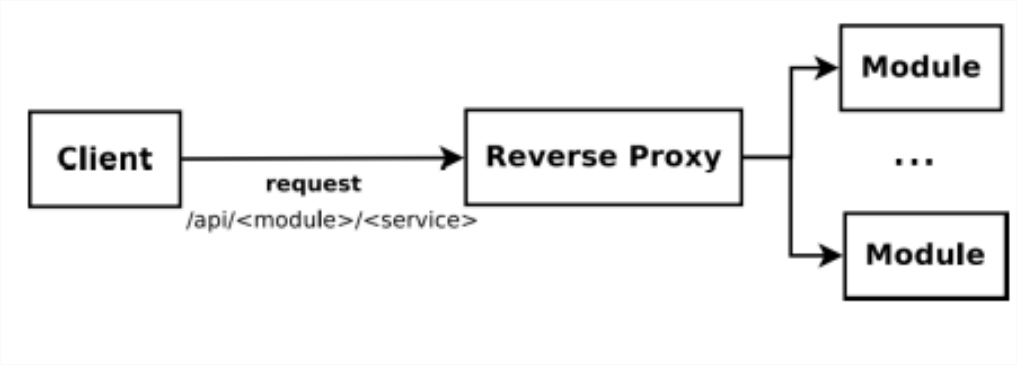
$\Rightarrow$ then an archive can be constructed including only needed files and independently of library or system header file.

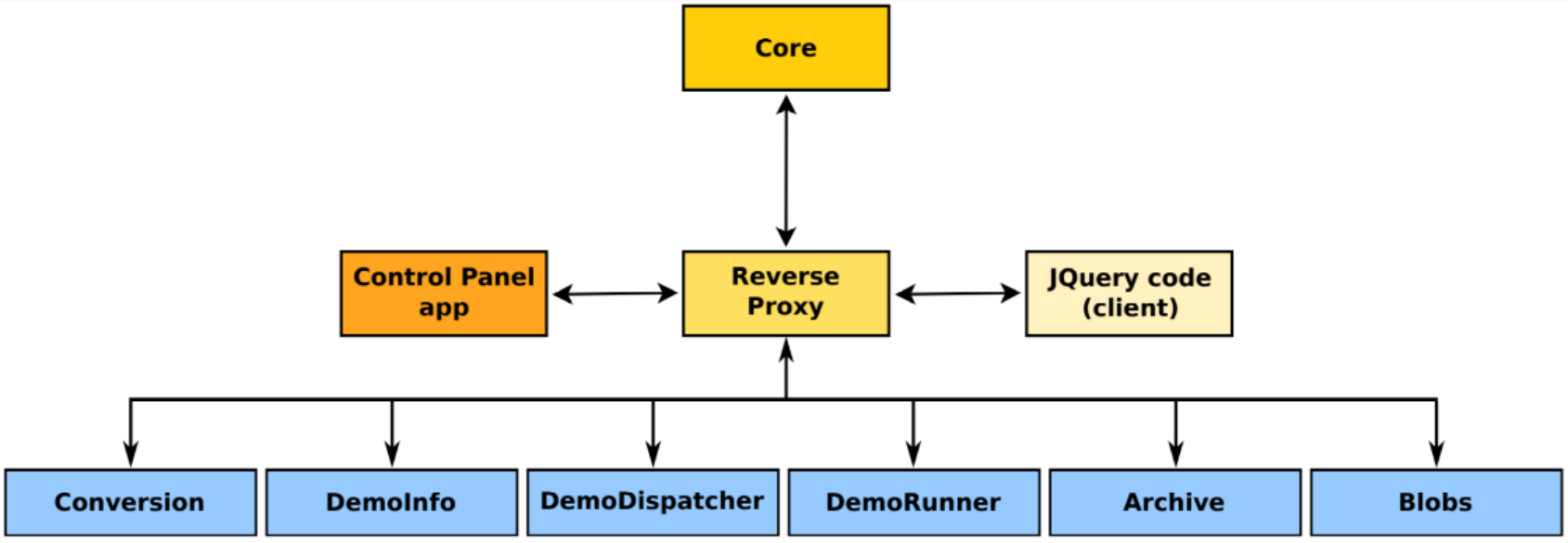
The IPOL Demo System

- Full architecture of microservices.
- Video demos are possible.
- Some servers with GPU.

$\rightarrow$ Extension to machine learning applications.

Demo architecture (initial one)





Demo architecture

Limitations

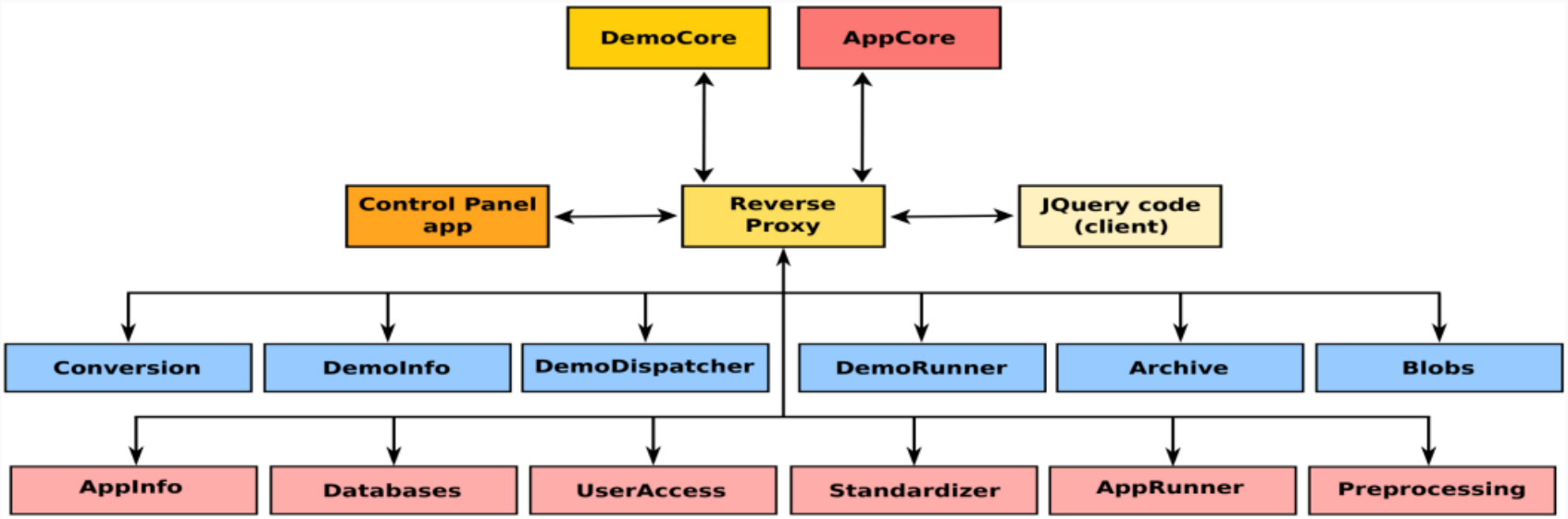
- OK for isolated demos.
- Demo is standalone. Does not share information with the others. Statelss
- Not well adapted to machine learning applications.
- New concept: application

Demo vs Application

- Execution time: demo starts and ends shortly. Appication never ends.
- The application sleeps when there is no activity. It might wake up when a new experiment is added to the archive.
- ML applications are more complex:
  - Pre-processing more complex (less structured).
  - Standardization of the data.

Demo architecture (current)

AppCore: same role as DemoCore, but controls the execution of applications.



Thank you for your attention