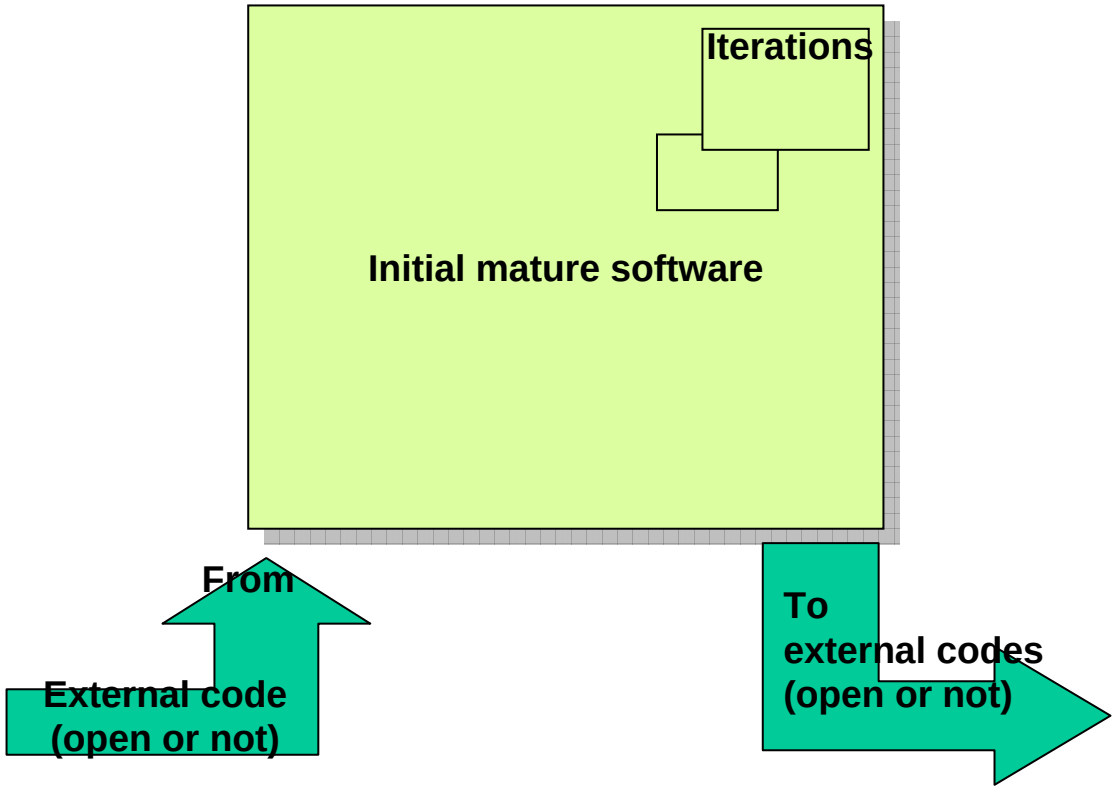
Marcel SOBERMAN
Resp. Valorisation & Partenariats industriels

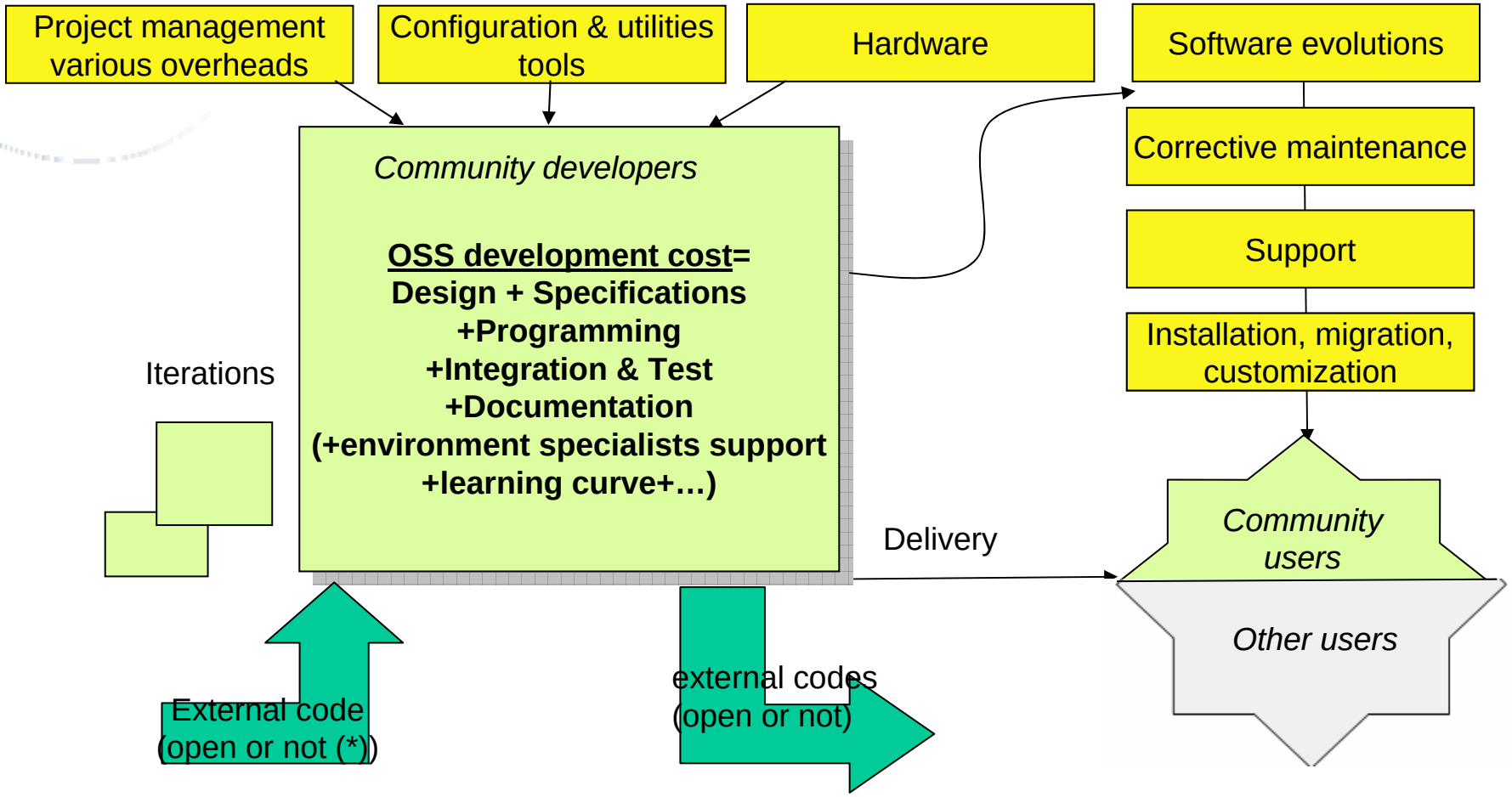
IN2P3
Institut national de physique nucléaire et de physique des particules

OSS RoI for the HEP community
Can we better ensure the sustainability of our software?

Large OSS development

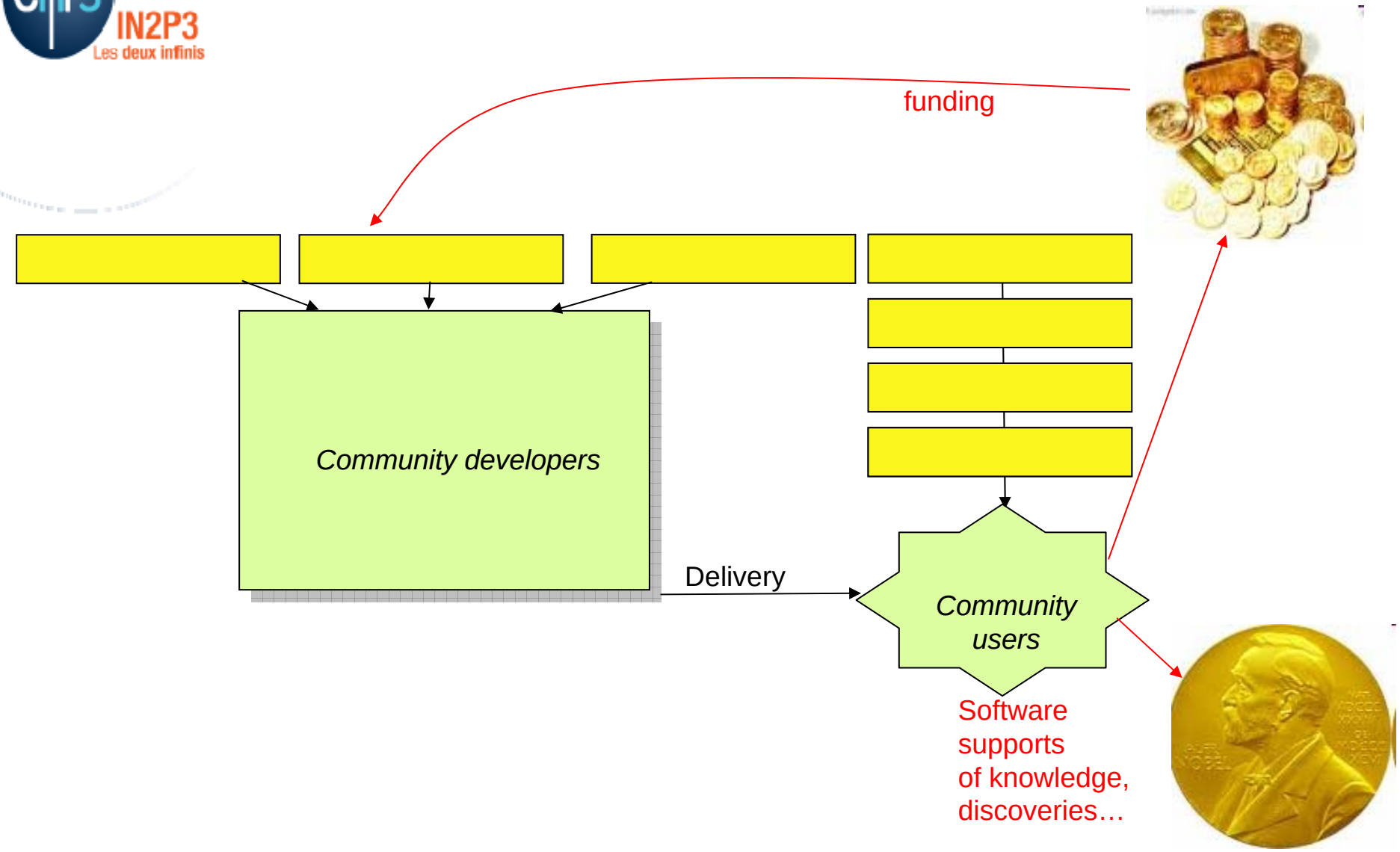


An ecosystem



*ex: http://www.opennebula.org/community:ecosystem

A virtuous circle inside the community



Problem!

- when the number of initial developers is going down without newcomers : maintenance, support, adaptation to new environments, evolutions are still there

Solutions:

- to find new public HEP funding, in general through R&D collaboration projects to add new functionalities
- to find new contributors
- to find payers: sponsoring (I have a dream…) or direct payers
- in the last case, three ways:
  - service offers
  - know-how licensing
  - software licensing

What is the cost of our software
How to measure software development c

- A lot of studies on development costs (Barry Boehm software metrics, COMOCO, function points…), but generally oriented on Information Systems development
- There are 3 great categories of software effort development estimators: function points, modules categorisation and comparison, and program sizes in lines of code
- additional factors: software complexity, language, quality of developers, duration of the project and of the learning curve, etc

Cost estimation

The most adapted indicator here is based on the effort in men-months to produce 1000 lines of reliable code given by the COMOCO model [source Wikipedia]:

$MM = a*KLOC^b$, with:

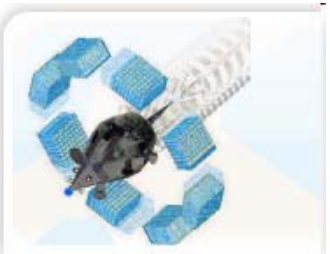
- a and b: coefficients -found in software development surveys- defining the complexity of the code and the diminishing returns in terms of code size, with a=3 and b=1,12 for intermediate code
- KLOC: kilo lines of codes (or KDSI: kilo delivery source instructions, and others acronyms)
- this includes design (=0,2 realisation), specifications (=0,3 realisation), programming or realisation (1), integration and testing (=1 realisation)

Cost estimation general example

- To have an idea of costs, we have taken the hypothesis of a software of 30000 lines → $MM=3*30^{1,12}=135$
- We have also compared with some examples like "Software Effort Estimation … _ Everything and the Mobile Software Universe.htm" giving for the same size: 97 to 185 MM; I have used 100 MM
- With the supposition of a mean cost of 0,8 young engineers and 0,2 seniors= 8 K€/month, the direct cost delivery of this software is: 8*100= 800 K€
- + overhead (management, communication, etc…): 7% (general rule in projects -> 856 K€)
- If we suppose a one year development, we have also to add:
  - investment for a new version: 3% (we suppose that the evolutions will represent each year 3% of the code)
  - maintenance and support: 10% (less than general rule for commercial software)

  → 800* (0,07+0,03+0,1)= 160 K€/year to assure software durability.

Real case: GATE



- Origin: a workshop organized by Irène Buvat in July 2001 in Paris, around the simulation toolkit developed in C++ by the Geant4 Collaboration
- An advanced opensource software developed by the international OpenGATE collaboration and dedicated to the numerical simulations in medical imaging
- Simulations of Emission Tomography (Positron Emission Tomography - PET and Single Photon Emission Computed Tomography - SPECT), and Computed Tomography (CT)
- Easy-to-learn macro mechanism to configurate simple or highly sophisticated experimental settings
- Key role in the design of of new medical imaging devices, in the optimization of acquisition protocols and in the development and assessment of image reconstruction algorithms and correction techniques.

GATE software survey: users profile

Users' profiles: positions

Computer scientist	3%
Imaging scientist / research engineer	33%
Professor / assistant professor	14%
Medical practitioner	5%
Clinical researcher	3%
(Medical) physicist	3%
Technologist	1%

Users' profiles: public vs private

From a public institute / lab	22%	86%
From a public hospital	18%	
University	46%	
From a private institute	5%	14%
From a private hospital	2%	
Private company	7%	

Irène Buvat - March 26th 2009 - 4

Users' countries

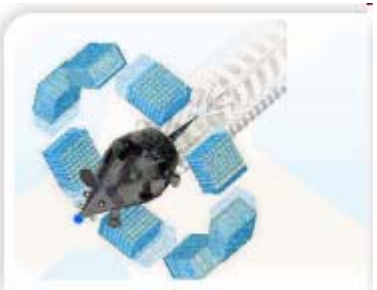
Europe	48%	
US	17%	33%
Canada	7%	
South America	9%	
Asia	15%	
Africa	4%	

25% not in the internal community

Irène Buvat - March 26th 2009 - 5

[source: www.opengate.org]

Gate effort



- 100 KLOC (GEANT4= 1 000 KLOC)
- # initial developers: 5
- # today evolutions developers (FTE): 2.5 to 5
- # wished support and maintenance: 2
- if we have to redevelop with expert developers, the realisation will take 24 MM → 48 to 60 MM including design, specification, integration and testing with a normal team (very far from the previous estimation based on COMOCO rules, but not taking into account some re-developments due to weaknesses in the first code, etc…)
- on the same basis than the previous evaluation, we need: 160*60/100= 96 K€ to assure maintenance and support of GATE.

Solution 1: new contributors: how external (to the consortium) users can participate?

- as new contributors:
  - specifiers and testers (but after?)
  - own developers, expanding the software through sub-communities
- participation through funding of evolutions, maintenance, and support
- by this way, the software can evolve, but the problem of maintenance and support is not fully solved

Solution 2: selling services

- Installation (payment on delivery)
- Support and corrective maintenance (often contracted support with recurrent funding)
- Customisation at various level (integration with existing software, adaptations…) (payment on quotes)
- and other consultancy services
- the client pays only the staff doing that with a limited profit

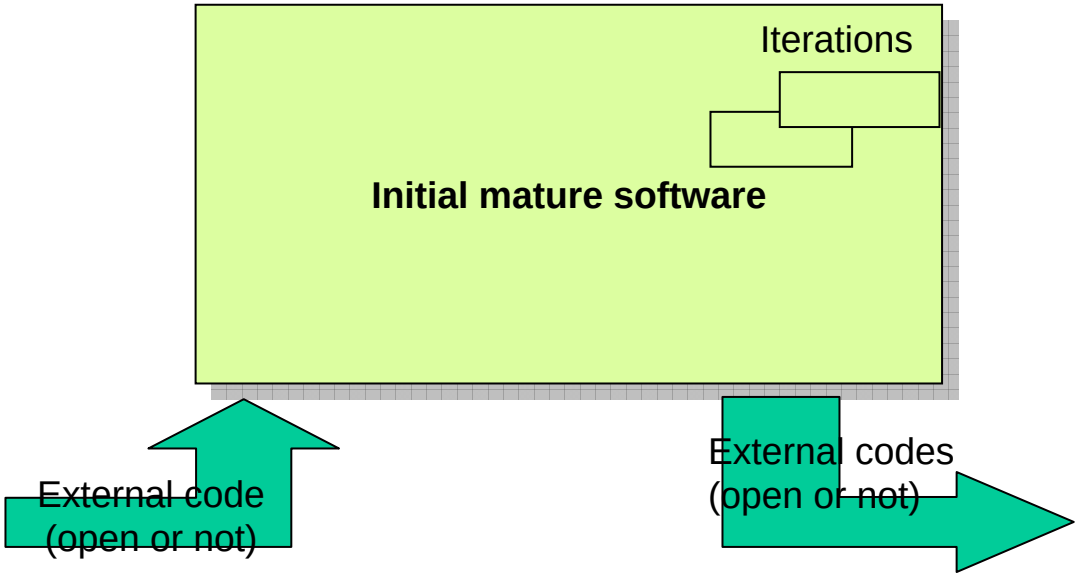
→ we cannot reimburse the cost of the software durability by this way

Solution 3: licensing

Three possible ways for licensing:

- Patents licensing – not possible in Europe unless embedded software
- Know how licensing:
  - Definition of know-how in IPR: secret, traceable
  - Know-how on OSS: limited, just on how to install it with high efficiency, how to use a complex software in difficult cases…
- Software licensing

The right license for open software licensing



- the choice of a license must be done carefully
- GPL is a restrictive barrier to integration with other software, please have also a look on LGPL, BSD, etc….
- select one with the obligation to send back all modifications, define a structure for the selection of the main ones and insertion in the software core; this will expand the community and keep the governance of the evolutions
- you don’t need to give your code to external uncontrolled users too early
- therefore, you can take time to select the right license, but you must have selected it and inform before authorising external users downloading

Free software is not ‘no fees’ software!!!

GNU GENERAL PUBLIC LICENSE Version 3, 29 June 2007

Copyright (C) 2007 Free Software Foundation, Inc. http://fsf.org/
Everyone is permitted to copy and distribute verbatim copies of this license document, but changing it is not allowed.

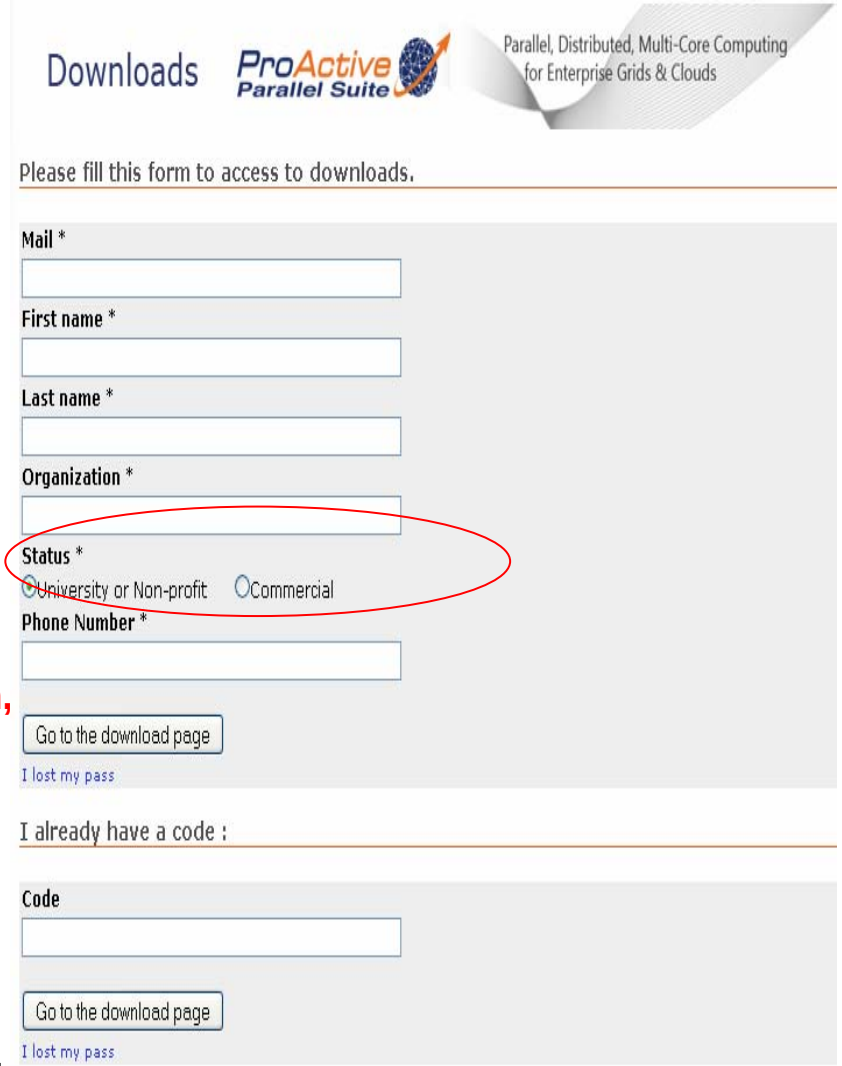
Preamble

The GNU General Public License is a free, copyleft license for software and other kinds of works.

The licenses for most software and other practical works are designed to take away your freedom to share and change the works. By contrast, the GNU General Public License is intended **to guarantee your freedom to share and change** all versions of a program--**to make sure it remains free software** for all its users…

When we speak of free software, we are referring to freedom, not price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software **(and charge for them if you wish),** that you receive source code or can get it if you want it, that you can change the software or use pieces of it in new free programs, and that you know you can do these things.

To protect your rights, we need to prevent others from denying ... For example, if you distribute copies of such a program, **whether gratis or for a fee**, you must ….



OSS+ complementary software licensing

- A core in OSS –free of charge- without warranty
- or limited to initial developers or contributors
- and –with fees-:
  - a reliable, maintained and supported version
  - additional components, in proprietary or open source
  - specific adaptations to each client

Dual licensing

- Two worlds, two licenses depending on the usage (in our case: basic research or not), or with complementary functionalities, warranty & support, etc…

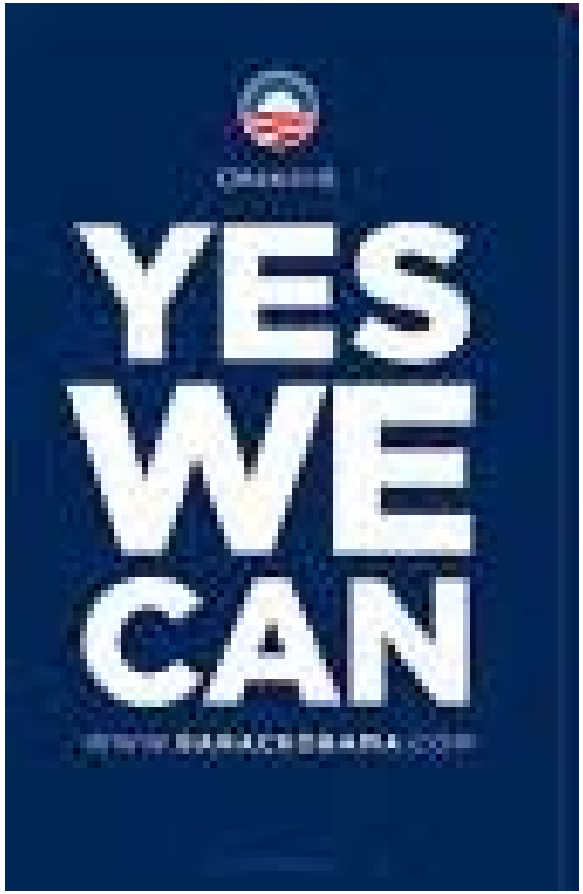
  Examples: ASTERISK, JASPERSOFT, MYSQL, PROACTIVE, QT, TALEND, VMWARE….
- see also: http://www.oss-watch.ac.uk/resources/duallicence2.xml
- Commercial rates on dual licensing given by software companies: 2 to 3%
- Software amortisement generally based on 3 years
- If we suppose 75% HEP users and 5 clients, price of a commercial license including support for one year=(856+160)*0,25/3/5= 17 K€ → not so expensive, very competitive! So you have now 17 K€*5= 85 K€ each year (the cost of a temporary contracted engineer to maintain and support the code for external AND community users)
- But first of all and since the first delivery, you must inform all downloaders that they have to decide on the choice of the right license, and why.

Solution 4: SaaS (Software as a Service)

- Why the user must install the software on his servers, add new modules, replace them for maintenance reasons, etc..?

→ SaaS

- and Saas is more and more offered in Clouds (or installed on a Grid)
- and this software, these grids and clouds, can be operated by the community
- but there are limitations, like grids reserved for academic research and some collaborative projects funded by EU – ex: EGEE-
- access fees (to not be unfair with companies offering also this use and to cover the cost of more equipments) is a good solution
- why not to add software licensing on usage – to reimburse at the same time our investment?, and particularly when we can have an added value with addition of services by specialists to build a package offer

Conclusion and questions

Can we ensure funding for a better sustainability of our software?