CSPA

Common Statistical Production Architecture

Descritption of the Business aspects of the architecture: business models for sharing software

Carlo Vaccari
Istat (vaccari@istat.it)

Index

- Costs categories
- Licenses available (also Open Data?)
- Business models
- Survey on tools and sharing
- SAB repository
- Other documents

Costs categories

- Criteria which may constitute a useful basis for decisions between different business models
- Buy_or_make alternative
- Open Source model vs Proprietary Software
- Total Cost of Ownership (TCO): when used in software TCO includes the sale price, any upgrades, maintenance and technical support, and training (or re-training)
- But time and frustration may be hard to measure!
- In the following costs categories to be taken into account

The purchase price of the system

For commercial software sum of licenses over the total period of usage; for OSS, obviously, there is no such purchase cost.

Many commercial packages (e.g. CRM systems for public surveys or advertisement) are quite expensive; on the other hand the financial budgets of statistical institutes are restricted and manpower is more easily available.

This may often be an important reason to go for OSS components rather than commercial ones, even if their functionality has to be enhanced for the tasks in question.

Cost of switching to the new system

Whether OSS or not, introducing a new piece of software requires a number of efforts to be carried out for installation / deployment / data migration (possibly including different hardware) and for training (of operators and users)

Lock-in issue for proprietary software

Cost of maintaining the system

Maintenance:

1 - Fixing bugs and 2 - Adding functions to the system

Bugs within commercial software should be (sooner or later) corrected by the vendor, the effort is paid for by the license; new functionality is added based on the marketing decisions of the vendor (<> customers intentions)

For OSS usually contributors do maintenance work for free and in most cases more rapidly than commercial vendors. However, in the limited marketplace of NSIs there are few companies and freelancers: work has to be done either by the NSIs IT staff or by paid companies

Cost of maintaining the system

This third category is the most important source of cost for software, and it is also the category where the “usage profiles” of specific statistics institutes are the most different

Again, if the budget of a statistical institute is limited, it might be easier to find manpower to maintain an OSS package than to pay high recurrent annual license or maintenance fees

Shared standards and choices between NSIs can greatly improve the ability to autonomously maintain software open

Cost of switching away from the system

When your system is no longer supported, neither by companies nor by communities, you must evaluate the cost of the migration of data to a new system

When your OSS is built on Open Standards, it is very probable that later OSS will use the same standard or at least be able to convert it

While most commercial vendors try to make data conversion to other vendors' systems as cumbersome as possible (and try to offer as many ways as possible to convert data from other systems into their own)

Introduction to Open Source

Once upon a time all software was “open” and “free”
rms at MIT in the first '80s collides with the first closed systems (PDP e Xerox)
1983: out from MIT, launches GNU (Gnu's Not Unix) – developments starts with developing tools like Emacs e gcc
1985: rms founds the Free Software Foundation

Note: free software, free as in freedom, not free as a beer!

1989: GPL General Public License

GPL License

GNU General Public License (GNU GPL or simply GPL), was created in 1988-1989 to protect the Emacs software.

Second version (GPLv2) released in 1991, together with a second license - the Lesser General Public License (LGPL) - less restrictive, useful for the C library and for software libraries that could substitute existing proprietary ones. LGPL permits the use of the software component in proprietary programs; using the ordinary GPL would restrict the component to be available only for free programs.

In 2005-2007 after a long debate: GPLv3, a release with important changes, regarded by many as too restrictive

Copyleft License

The most important characteristic of the GPL license is the copyleft which aims to preserve the freedom and openness of the software

GPL is aimed at giving the end-user significant permissions, such as the permission to redistribute, reverse engineer, or otherwise modify the software, but the end-user must redistribute any modification under the same license (so called “viral” license)

Apache License

The Apache license is released by the Apache Software Foundation, the organization that manages the most used web server in the world (see) and many other projects.

The Apache license requires preservation of the copyright notice and disclaimer, but it is not a copyleft license, i.e. it does not require modified versions of the software to be distributed using the same license.

This license allows the use of the source code for the development of proprietary software as well as free and open source software.

BSD License

BSD licenses are a family of permissive free software licenses, originally used for the Berkeley Software Distribution (BSD) Unix developed in the University of California (Berkeley) (also MacOsX IS a BSD!)

These licenses have fewer restrictions compared to GPL, putting works licensed under them relatively closer to the public domain, i.e. content that is not owned or controlled by anyone – "public property"

Open Source Software (OSI)

1988: Open Source Initiative defines “open source”:

- Free redistribution (for free or not)
- Freedom to consult the source code
- Derived works approved
- Integrity of the author's source code
- No discrimination against persons or groups
- No discrimination against fields of application
- The license must be distributed (no additional license)
- The license may not be specific to a product
- The license must not restrict other software
- The license must be technology neutral

European Union Public License

European Union Public Licence (EUPL) is a free software license created by the European Commission.

First approved in 2007, current version (1.1) 2009

The license is available in the 22 official languages of the European Union

The EUPL v 1.1 is OSI certified as from March 2009.

The main goal of the EUPL is its focusing on being consistent with the copyright law in the 27 Member States of the European Union, while retaining compatibility with popular open-source software licenses such as the GPL license

Possible business models

Maintenance (for problems which in most cases do not show up in the production environment of the supplier himself) and doing paid development for someone else's requirements is not "the core business of statistical institutes"

So, how to support development and training on OS software?

In the following different ways to use and improve OSS for (statistics) administrations throughout Europe

Possible business models

Barter model

Bartering is one archaic form of trade: Exchanging work or goods on a non-monetary basis.

This model is used quite frequently in environments where equivalent partners cooperate without any explicit payment.

Support & Training: Each participant provides support and training for those parts they contributed.

Possible business models

Co-development

This mode of cooperation does development in a (loosely coupled) project financed either by a consortium of statistical institutes or by an international entity (e.g. Eurostat)

The explicit payment of work allows the involvement also of external companies for specialised tasks

Support & Training: As with barter, each participant provides support and training for those parts they contributed

Possible business models

Freeware

Freeware may be obtained and used without payment, similarly to OSS. However, the main difference is that freeware doesn't give access to the source code – hence, no further development is possible.

If the software fulfills its requirements, this is fine; otherwise freeware is not applicable.

For freeware there is usually no support or training available

Possible business models

OSS development / support: 3 possible cases

- No original support
- Local contracted support: (similar to Co-development) set up a follow-up OSS project with interested partners doing further development and then giving support. Usually, this will work if the interested administration will sponsor the project.
- In-house support: similar to Co-development, with the difference that work will be done by your internal employees. This is the kind of OSS project many large software companies are doing