On the sustainability of the SDC software tools

Daniela Ichim and Luisa Franconi
Istat, Division for Information Technology and Methodology
ichim@istat.it, franconi@istat.it

1. Introduction

Till the present time Eurostat has been in charge of managing the confidentiality of European statistics, both for microdata and tabular data. We just mention two examples to give an idea of the amount of the work involved. Eurostat is currently preparing the microdata files for research for all the surveys mentioned in Regulation 831/2002 and for (mainly all) member states. Eurostat has dealt with the disclosure protection of all the tables released at European level stemming from the whole system of surveys dealing with structural business statistics for all member states.

However, as we experienced an increase in the number of statistical products, in the number of surveys, in the number and complexity of tables (and microdata) and in the number of member states producing all such statistics, Eurostat found it more and more difficult to manage all such burden. It was clear that member states needed to share responsibility for privacy protection of their own statistical products contributing to European results and needed to share the burden for such work. This is undoubtedly a big change of perspective that has already taken place in certain areas. The first examples of this "new course" are the regulation concerning particular aggregates of foreign trade statistics and, lately, the regulation on structural business statistics (entered into force in April 2009). Before such date Eurostat was in charge of applying the confidentiality rules for each member state and for each table; after 2009 member states will be required to provide Eurostat with protected structural business statistics tables. This is not just a mere example but a significant change in view and it will possibly be followed by other surveys therefore making member states more active in the area of confidentiality for European statistics.

If, on the one hand, such change, implying redistribution of work, requires a big investment and a large effort from all NSIs in Europe, on the other hand this will hopefully lead, in the medium term, to a widespread increase in the level of knowledge and expertise in the field of confidentiality in Europe. Once a procedure has been learnt in a survey it could be applied similarly to other surveys leading to a virtual circle of innovation inside each National statistical institute. NSIs will be confronted, sometimes for the first time in certain areas, with a new process and new tools to implement to their policy in terms of confidentiality. The tools freely available at the moment are the ARGUS software (see 1) and the two R packages (see 2, 3 and 4). The adoption of a tool by an NSI implies an investment in knowledge and big costs in terms of staff training. The question of sustainability of the adopted software then becomes a key issue. Sustainability not only in terms of capacity to endure through time, but also in terms of capacity to cope with new situations where a high number of users need to apply to their survey production cycle a new product. In fact, on the one hand the software should be maintained and upgraded constantly to be adaptable to future and possibly more complex situations derived from new needs. On the other hand, the "forced" enlargement to a high number of potential users and to new surveys and statistics implies an exponential increase in the number of questions to be answered and problems/bugs to be solved.

We claim that a way to reach sustainability is to move towards an open source software. This will enable both the continuous changes and modifications to adapt to new situations and the sharing of the burden of problem solving among a larger community.

The aim of this report is to analyse, starting from the current situation, possible ways of reaching sustainability providing pros and cons for each solution and identifying several steps to be taken to reach the aim. Section 2 presents the current situation for NSIs as far as use of SDC software tools are concerned. Section 3 analyses possible costs of not achieving an open source software. Section 4 identifies an organisational solution for an open software and a roadmap to reach sustainability and more knowledge sharing. Section 5 presents some conclusions.

2. The current situation

The current situation was well described in (5). There are mainly two software tools applied by the SDC community: ARGUS twins and two R packages.

2.1 ARGUS

ARGUS is a free software that may be downloaded from the CENEX/CASC website, http://neon.vb.cbs.nl/casc/index.htm. It has two modules, μ and τ ARGUS. The first one deals with microdata SDC problems while the second one, τ ARGUS, deals with tabular SDC problems. Both ARGUS twins implement several methods of risk assessment and disclosure limitation. Anyway, it should be observed that, for the application of some particular protection algorithms, τ ARGUS might need a non-free optimisation licence.

ARGUS is currently developed and maintained by one or two persons at Statistics Netherlands. Anyway, ARGUS was mainly developed as part of several European projects. Besides Statistics Netherlands, some other European National Statistical Institutes and universities contributed to the development of ARGUS, e.g. ISTAT (Italy), DESTATIS (Germany), ONS (United Kingdom), Rovira i Virgili University (Spain), University of Ilmenau (Germany) just to mention a few. One of the main aims of ARGUS (project) was to develop a general applicable de facto standard tool for the disclosure control of microdata and tabular data. A coordinated approach for the SDC topic (best practice action) was also aimed. The participation of several European NSIs guaranteed the general applicability of ARGUS.

In a certain sense, ARGUS is a user-friendly software. No programming skills are required for the SDC problem itself. However, ARGUS depends on a system of metadata that, due to the nature and complexity of the tackled problems (such as linked hierarchical tables), might be cumbersome to manage. For this reason some other software knowledge/application may be useful in processing the ARGUS inputs and outputs. Moreover, ARGUS is platform dependent, i.e. was designed only for WINDOWS.

ARGUS was built on the competence, expertise and reputation of people working for many years in the SDC field in official statistics. Although the actual implementation of the majority of the modules was completely in charge of Statistics Netherlands, such NSI is not the only responsible for the algorithms implemented in ARGUS. The partners of the above mentioned European projects were in charge of developing, implementing and testing the ARGUS modules. The participation of several National Statistical Institutes to the ARGUS development allowed its testing on real case studies in official statistics. The direct involvement of NSIs, as well as Eurostat, made it possible to tackle several specific issues stemming from peculiarities of specific surveys. Also, as a result of such involvement, a coordinated development was achieved, with predictable results.

It should be noted that ARGUS is still under development. First this is due to the limited resources allocated to the software implementation (Statistics Netherlands). Second, this is due to new

approaches/methods/algorithms in the SDC field that need to be implemented and tested to solve more and more complex problems in official statistics applications. Improvements in ARGUS are continuous, even if the ARGUS updates/releases are not continuous as well. These improvements are possible grace to the ARGUS modularity, even if the implementation is in charge only to Statistics Netherlands. The efforts spent by Statistics Netherlands should be highly appreciated. Moreover, the competence and experience acquired during the European SDC projects should/could be exploited.

Both μ and τ ARGUS have a documentation manual. The partners of the European SDC projects contributed to the ARGUS documentation. The two manuals describe the full SDC process in a step by step manner. Anyway, improvements are always possible and welcome. Instead, the help of ARGUS is almost absent. Several error messages are even written in Dutch.

Other characteristics of ARGUS were discussed in (5). Its main "drawback" is a feature of any proprietary software: there is no possibility to check, control, modify and adapt to the user needs. It should also be mentioned that the existence of only one maintainer could create some difficulties. For example, what happens if the (unique) web server breaks down? Of course, the simple existence of mirror sites would solve this latter issue.

In a different perspective, another drawback of the current software architecture is that ARGUS was mainly designed for the statistical disclosure control problems encountered in official statistics. Even if it could be used in other fields as well, the SDC problems approached up to now were derived from Official Statistics. For the moment, the contributions to ARGUS development were given only from the Official Statistics field. This is mainly due to (and grace to) the European projects that sponsored the ARGUS development.

2.2 R

The second tool discussed in this document is the R software, especially the two packages sdcMicro (see 4) and sdcTables (see 3). These two packages are implemented and maintained by one developer from University of Wien.

R is a free and open source software, available at www.r-project.org. The two SDC packages (and other required or suggested packages) may be downloaded following the link "PACKAGES".

Independently on the SDC packages, R is not a user friendly software and an initial programming investment is required. This investment would be later awarded by the immense flexibility of R. Using one of the many GUI developing tools (see http://www.sciviews.org/_rgui/), an interface to R, or, at least to the SDC packages, could be built. In such cases, several problems of the proprietary or non-free software could be partly inherited. For example, the interface maintenance should be managed. The sole existence of an interface is not a constraint to using it.

Like any other open source software, R is copyrighted and distributed under a licence which gives the licensee a great amount of freedom in the area of further development (modifications, enhancements, localisation, peripherals, integration, bug fixes and re-distribution). Namely, R is available under the terms of the Free Software Foundation's GNU General Public Licence, see www.gnu.org/licences/#GPL.

By construction, R has a modular architecture which greatly improves its flexibility. Even from the point of view of the metadata, R has noticeable advantages. There are some standard objects (data structures, methods, etc.) already created/implemented in R. These objects cover the common data

structures used in statistics. Anyway, the user might create the new objects needed for any particular application.

The source code of R and its packages are maintained at www.r-project.org. There are also several mirror sites which ensure the continuous availability of the software. Just for historical and compatibility reasons, the older versions of R and each of its package are also maintained. Consequently, the stability is guaranteed, in the sense that migration to new versions is not imposed. The availability of the source code for several operating systems (Linux, Windows, and Macintosh) guarantees its large diffusion in the statistics community. Moreover, the source code free availability guarantees that no secret back-doors exist. Generally speaking, the open source software uses transparency to increase quality.

The source code of R and its packages are continuously updated. Each year there are two major releases of R. The new contributions to R generally take the form of a new package. Due to applied copyright licence, the updates/corrections and bugs fixes of existing packages are "in charge" to the initial author. For example, any (free and open) contribution to the two SDC packages is managed by University of Wien. Only for major contributions, the list of authors could be enlarged. Anyway, only the initial author has the possibility to decide for such an authorship extension.

There is a consortium that manages the development of the source code of R. The packages are only modules that could be "attached", according to the user needs. Each package is developed/documented and maintained by its authors. Only the authors are responsible for the implemented code and for its testing.

An important characteristic of any free and open source software is the users' role. Generally the users are simultaneously developers and testers. The users are the persons knowing at best their needs, having the best possible and detailed knowledge of their applications. Consequently, their developed software should guarantee the applicability in real cases. The implemented solution could be suitable for some particular case-study, but people working in the same field, generally need similar solutions. The opportunity to test the software on real problems is another important advantage. Indeed, the sole implementation of some algorithms does not guarantee their applicability in practice. Only an exhaustive testing on real data could contribute to the large diffusion of the software. The software implemented without any testing is generally considered less credible. Moreover, at least in the SDC field, the risk of disclosure should be correctly estimated and the declared protection should be really achieved. In order to test the software in real SDC applications, the access to confidential data is a key point. It follows that the participation of SDC people in the SDC software development would be only a positive input. Of course, the development shouldn't be restricted to SDC people.

The main drawback of any free and open source software is its predictability. Generally, it is impossible to impose or to predict the contributions that might arrive. Not even an agenda of the software release could be well defined. The success of any open source software depends on the degree of involvement of the interested community. If needed, the community of users/testers could be enlarged by better addressing its needs.

Documentation and help play another important role to the success of any open source software. Since its beginning, R was built as a well documented software. Lately, with the exponential increase of the number of packages, those packages seem to be less documented. Moreover, their corresponding on-line help system doesn't seem to maintain the same quality standards. The lack of quality of the documentation system could promote inappropriate solutions or concepts. The documentation and help of the current versions of the two SDC packages could be greatly improved.

3. Cost of not migrating ARGUS to an open source

This section analyses possible costs deriving from the maintenance of the *status quo* i.e. the cost of not migrating Argus to open source.

Short-medium period: Lack of resources to provide answers under pressure.

The development of the ARGUS software is maintained and built under European projects. In such projects the testing of the software is achieved by project partners. When the list of "to do" things becomes too long the resources allocated by Statistics Netherlands cannot cope with all the changes and fixings asked by users and some problems/aspects cannot be developed (and users get frustrated!). Just to give an example, in the near future the developers will undergo a period of extreme pressure when, due to the new structural business statistics regulation mentioned in section 1, member states will be asked to provide Eurostat with protected tables. This situation will appear every time a change in the law will move the responsibility of undertaking confidentiality protection to a member states. Then, all the questions arising from this exercise will be on the shoulder of the small developing team. As such trend where member states will be asked to provide protected tables or protected microdata will possibly cover more and more surveys, such lack of resources could create serious management problems.

Medium-long term period: possible shortage of CBS staff devoted to software development

As for the current situation only Statistics Netherlands is able to modify the software in any way: relying on a single institution for what has the ambition of becoming the European standard could be seen as a hazard from some NSI. Moreover what if the expertises built so far at Statistics Netherlands would for some reasons move/leave their job? All the investment made in several years of European project will vanish abruptly.

Finally there is a last element that need to be taken into account: the difficulty in modifying the software and the frustration in waiting to see the changes implemented may in the end lead NSIs to the creation of their own software tools to solve their particular needs reinventing each time things that have already been implemented. This would be a highly inefficient way to manage the problem.

4. Bridging the gap

In this section a way to move forward is described. The goal is to change the current situation to ensure the sustainability of the SDC tools in the future. The proposed solution is a hybrid one. Indeed, it should inherit the main strengths of the previous two cooperation models (R and ARGUS). At the same time, the solution should reduce as much as possible the main drawbacks of those models. Issues of communication, coordination and control will be discussed.

The current situation is summarized in Table 1.

Table 1: Features of the available software tools in the current situation

Feature	**ARGUS**	**R packages sdcMicro and sdcTables**
Possibility to check/control/modify/adapt	NO	YES
Coordinated development	YES	NO
Predictable results	YES	NO
Development agenda	YES	NO
SDC people involved	YES	NO
Documentation	YES	YES/NO
Help	NO	YES/NO
Modular architecture	YES	YES
Extensibility	YES	YES
Platform dependent	YES	NO
Unique maintainer	YES	YES
Personalisation	NO	YES
User-friendly	YES/NO	NO
Programming skills required	NO	YES
Free	YES	YES
Open source	NO	YES
Designed for official statistics	YES	NO/YES
Mirror sites	NO	YES
Consortium	YES	NO
Test reports	YES	NO
…	…	…

Several steps helping a smooth migration to a new open source SDC tool might be the following:

4.0. ALWAYS KEEP IN MIND THAT ANY MIGRATION/CHANGE HAS A COST!!!

4.1. USE ARGUS ARCHITECTURE

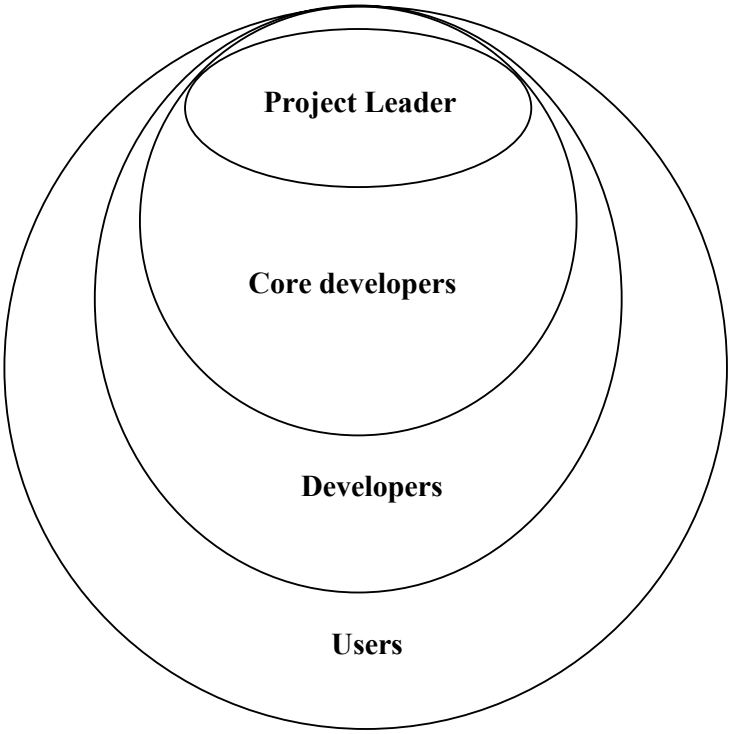
The proposed solution is based on the current architecture of ARGUS. This choice is due to the fact that ARGUS is the software tool most known and applied by the SDC community in Official Statistics. Why a new cooperation model should be designed when the existing one could be easily adapted and then used? In this way no effort should be spent in designing new architectures or in creating new collaboration networks. By adopting the existing architecture, the SDC community would spend its attention on incremental development, rather than on design discussions.

One of the main features of the ARGUS architecture that should be maintained is its modularity. A flexible development and a high degree of personalisation would be then ensured.

4.2. CREATE AN ORGANISATIONAL STRUCTURE

The proposal is to create the governance structure illustrated in the Figure 1.

Figure 1: Organisation structure



First of all, a governance structure is needed because it is the only way to guarantee a coordinated development, as well as the predictability of the results. Second, only an experienced governance structure might identify the important projects and establish their priorities. Third, many open source software have an organisational structure that prevents attempts to fork. Probably the most well known examples are Linux and R itself.

Project leader (PL)
One individual responsible for design decisions concerning the overall direction of the project. The project leader should be also responsible for defining and delegating responsibilities to core developers.

Core developer (CD)
An individual with distinctive responsibilities defined and delegated by the project leader. The core group should comprise qualified developers who have demonstrated good knowledge and good skills. They should be responsible for creating most new functionalities. In order to join the core group, candidates (developers) should prove their competence and enthusiasm.

Developer (D)
An individual who has demonstrated a desire and ability to contribute to the project. A developer role gives write access to the source code and is granted by the project leader. Developers should commonly contribute bug fixes and small enhancements.

User (U)
The tool users are actively involved in the project by testing and debugging the product, identifying and reporting bugs, suggesting new features and participating in the mailing lists.

The project leader role is the core to the software success. In particular, his capabilities in attracting, motivating and coordinating a team of skilled developers should play an important role. The major reason open source projects fail is a lack of user-contributions to do the work. An important task of the project leader would be to avoid that the issues (projects, problems, questions) become dormant. The project leader should also monitor the overall activity of the software project. Suitable monitoring tools should be made available for this task. The project leader should be the first responsible for the strategic decisions.

The allocation of the other roles should be dynamic, controlled by the project leader and based on participant interest and demonstrated contribution. If a role is not being actively filled, then a keen and able participant should be drafted in to fill the vacuum position to be taken.

To the success of any open source software, the communication mechanisms (mailing lists, forums, web sites, etc.) have an extremely important contribution. First this allows the access to updated information. Second, the feedback mechanisms should allow/encourage people to share the problems and solutions. Indeed, when as user/tester has a problem, he may want to ask the question to a geographically distributed community. The developers should (but they are not obliged to) guarantee the updates/the bugs/the corrections. No one could oblige the developer to modify the code, but the price to pay would be his reputation, especially when the bugs were posted on publicly available forums or mailing lists. In case the developer wouldn't solve the given problem, some other user/tester could have at least an approximate solution or could indicate some reference or starting point. It should be stressed again that the project leader should have the responsibility to manage (solve, delegate, etc.) the unsolved issues.

The infrastructure should be strongly facilitated by Internet connectivity as a shared medium of communication. The infrastructure should comprise mailing lists, issue tracking, a homepage and a helpdesk.

4.3. OPEN ARGUS

ARGUS should become an open source software. This is probably the key point to ensuring the ARGUS sustainability. Of course, this decision depends only on the Statistics Netherlands willingness to open the source code. In an initial migration stage, the code could be opened at least to the partners of the European SDC projects.

The European Parliament has voted resolution A5-0264/2001 that "Calls on the Commission and Member States to promote software projects whose source text is made public (open-source software)". Consequently, the Eurostat sponsorship for an open-source SDC software should be expected. For example, Eurostat could delegate the initial project leader and the initial team of core developers.

Until the complete migration to a new SDC tool, the older versions of ARGUS should be available at the CENEX web site.

4.4. MIGRATE ARGUS TO R

An option could be the usage of R interfaces to a compiled code. This option was already discussed in (5). It is not a real migration and it would only inherit the drawbacks of a proprietary software.

Another option could be the translation of ARGUS in R. Partly, the algorithms were already translated in the R packages sdcMicro and sdcTable. But the application of those packages in real problems was not yet reported. Probably the best solution would be the translation of the ARGUS code in R. A small team of ARGUS and R experts could work together on this project. The case studies where ARGUS was applied could be used for testing the compatibility of the new R code with the ARGUS code.

It should be noted that the SDC tools are (or should be) used by people involved in any stage of the dissemination process, e.g. survey experts. Generally speaking, the SDC tools should provide methodological support for data dissemination, which is deeply related to the survey production. That's why it would be recommended to create an interface for the newly created SDC tool. The interface and the core part of the software should be maintained and updated by the project leader and the core developers. An interface and an open source software are not two exclusive options.

4.5. UPDATE THE DOCUMENTATION AND HELP

Obviously, the documentation and help should be modified and adapted to the new SDC tool. This is a crucial step for promoting a new software. The documentation and help should be prepared by the migration team together with several SDC experts.

4.6. TRAINING COURSES

A series of training courses should be foreseen. This is a fundamental action for the success of any new software tool.

5. Conclusions

We have seen as for structural business statistics we are moving from a situation of "monopoly" by Eurostat of making confidentiality protection to a situation of shared duties among member states; each for their own data but leading to a common goal: protected European statistics. This move has taken place also for some statistics in the foreign trade and possibly represents a change in the division of responsibility and workload among Eurostat and member states as far as confidentiality is concerned.

As far as the software ARGUS for statistical disclosure control is concerned we are now in the situation of "monopoly" of Statistics Netherlands (CBS). Although the software is freely available, it is not possible to make changes or to modify only slightly a particular feature.

Moving to an open source software with a clear organisational structure with different levels of permission as described in this paper would allow the sharing of duties among different entities. We believe that making the ARGUS software an open source software would be extremely beneficial for spreading the knowledge of the software itself and for improving the creation of expertise in the area of statistical disclosure control as well as for developing more and more innovative solutions.

6. References

1. ARGUS software, available at http://neon.vb.cbs.nl/casc/
2. R Development Core Team. R: A Language and Environment for Statistical Computing. R Foundation for Statistical Computing, Vienna, Austria, 2008. URL http://www.R-project.org. ISBN 3-900051-07-0
3. Bernhard Meindl (2009). Linking complementary cell suppression and the software R, New Techniques and Technologies for Statistics, Brussels, 18-20 February 2009, available at http://epp.eurostat.ec.europa.eu/portal/page/portal/research_methodology/documents/S18P2_LINKING_COMPLEMENTARY_CELL_MEINDL.pdf.
4. M. Templ (2008). sdcMicro. Manual and Package. Version 2.5.1. Statistics Austria and Vienna University of Technology, Vienna, Austria.
5. M. Templ (2008). Making SDC-tools better usable by NSIs: Sustainability of the Argus Software, Research Report CS-2008-5 (available on the ESSNET SDC internal web site).