THE PROCESSING ENVIRONMENT BEHIND A STATISTICAL PROGRAM

Claude Poirier
Statistics Canada, Tunney's Pasture, Ottawa, Ontario K1A 0T6

Key Words: statistical functions, processing tools, modular systems, foundation software

ABSTRACT

In developing a statistical program, a national agency must also develop the related technical environment, ideally including common tools. This paper examines the strategy of developing such an environment by considering the scope of the common tools, the user requirements, the software components, the budget issue and the management pressures. In this context, past lessons should be considered in identifying the set of robust, common and standardized methods which would most likely satisfy a wide range of internal applications. Furthermore, the environment should be reliable, simple and flexible enough to be expanded throughout the years. Given that the end-uses of the tools determine the success of such a project, the promotion of the resulting products is addressed. Statistics Canada's positive as well as negative experiences form the basis for this paper.

1. INTRODUCTION

The development of a statistical infrastructure requires a framework covering both the technical environment and the statistical functions. In such a context, the framework must consider the cost, the robustness and the expandability of the end product. Reusable modules are key elements to achieve a cost efficiency and expandability. For this reason, statistical agencies should consider generalized modules as the basis of their statistical infrastructure. Statistics Canada has a history of generalized systems development which dates back to the 1970s. During those 30 years, Statistics Canada has developed several "core" generalized systems. These include systems for the survey functions of sampling, edit and imputation, estimation, autocoding, disclosure control, record linkage, and time series analysis. A brief description of each of these systems is given in Statistics Canada (1999). Although a few customized systems have focused on the functions of survey tabulation and data analysis, there have not been any that have yet reached a generalized level at Statistics Canada. More information on the overall strategy for generalized systems development at Statistics Canada are provided by Outrata and Chinnappa (1989), Turner (1994), and Kozak, Poirier and Kovar (2001).

The development of generalized systems cannot be achieved in a year or two. They are rather the result of decades of effort which most likely starts with theoretical research activities, feasibility studies, and the development of prototypes that are built for specific projects. Then, several years of fine tuning and evaluation are required before one can think about a generalized version, and even then, not all systems will reach that stage. There still are numerous systems designed at Statistics Canada which are more customized in nature. Some may be good candidates for generalization. A good example is the Canadian Census Edit and Imputation System (CANCEIS) based on the Nearest-Neighbor Imputation Methodology (NIM). While the system was initially designed to process demographic data in the Canadian Census, the developers have been working on a more general approach to process both qualitative and quantitative variables. Further details are provided by Bankier et al (1999).

2. USER REQUIREMENTS

For several decades, many national statistical agencies had let survey managers initiate system development without exerting too much control. When agencies had to face budget cuts, they started to streamline technologies and functions. Many major redesigns were initiated in the past 20 years. Examples are the development of a standardized economic processing system in the U.S. Census Bureau, the modernization project in the U.K. Office for National Statistics, and the development of generalized systems in Statistics Canada. The first step of such projects is the identification of goals and objectives. A public institution like Statistics Canada doesn't develop statistical tools with the intention of selling them. Rather it targets the automation of its current processes by acquiring existing commercial products whenever they exist, and by developing some others only if needed. In the latter case, the developer requires a clear understanding of the user requirements. In practice, this is achieved through a documentation process in which both the user and the developer actively participate. Hopefully, the developer will have access to well established networking resources through which similar

requirements from other projects will be identified. The principle here may be to combine the requirements of a few projects together and to take advantage of common efforts. This means that challenges can be assigned amongst several developers and development time can be reduced resulting in a more timely and robust product for the investment.

In pooling requirements together, the agency should take the position of focusing on the use of a single data structure for most of its systems. For example, at Statistics Canada, executive decisions have been steering development toward the use of SAS as that data structure since the 1990s. There are several reasons for this, including the fact that there is already widespread usage of SAS at Statistics Canada and thus a broad knowledge base already exists. SAS also satisfies the user requirements for portability across the different computing platforms in use within the bureau. Note that before SAS, Oracle was the preferred data structure but it was soon realized that despite its technical advantages, Oracle was not always suitable for computationally-intensive statistical functions.

As opposed to commercial developers, the mandate of statistical agencies is not to develop fancy systems but to support statistical programs. They must satisfy the internal needs by providing a simple and robust infrastructure. This means that the focus should be on the processing aspect in order to make the systems reliable, efficient, affordable, robust, require minimum maintenance, cheap and coherent with the infrastructure. While the ease of use and reuse clearly competes against the above features, from the user's point of view it is important that the learning curve be short in order to speed up development of survey applications. User-friendliness of systems is one of the objectives to be strived for in reaching the overall goal. The use of graphical user interfaces, modular components, and efficient input-output functions are seen as the primary means of achieving this objective. The survey managers are responsible for balancing these objectives when planning the development activities.

3. THE QUALITY ASPECT

Statistics Canada has developed quality guidelines which target statistical products (STC, 2003). The document mentions that "*A significant feature of the management of quality is the balancing of quality objectives against the constraints of financial and human resources*". The goal of a software development project should not be the maximization of quality at all costs, but the achievement of an appropriate balance between the quantity (the number of features) and quality.

A multidisciplinary project team is required to conduct a system development project. The project team makes the many decisions necessary to ensure that there is an appropriate balance between concerns for quality and considerations of cost. The team should consider all of the dimensions of quality, even if some are conflicting, while identifying the optimal system design. The following paragraphs describe the various dimensions of quality according to STC (2003). While these dimensions primarily apply to statistical products, they can be adapted for system development.

Relevance: The relevance of a processing system reflects the degree to which it meets the real needs of the client in terms of functionality. In the case of multiple clients, relevance is influenced by their varying needs. Within the context of statistics programs, the relevance of systems is the key dimension of the quality. A system may be reliable, robust, powerful, user-friendly and well documented, but if it does not address the needs of the statistical program, it loses its quality. The challenge of an agency is to weigh and balance the needs of current and/or potential users with the resource constraints. This exercise requires the translation of user needs into program approval and budgetary decisions. To achieve relevance, an agency should care about system diversity. In other words, it should try to keep the duplication of functions as low as possible across systems, and prioritize development initiatives which address system gaps.

Accessibility: The accessibility of a system refers to the ease with which users can work with it. This includes items like platform constraints or portability, installation protocol, flexibility, adaptability, integration, power, userfriendliness, documentation, support resources, etc.

Interpretability: The interpretability of a system reflects the level of information that is made available to help users in understanding both the system input and output. Systems require some sort of data dictionary or metadata to ensure interpretability, regardless of its complexity. Other elements which contribute to the interpretability include system documentation, record layout, on-line help, diagnostic reports, error and warning messages, etc.

Coherence: The coherence of systems refers to their level of standardization. Coherence is the key element of integration. It helps to bring together systems to form a suite of systems which can be run one after the next to ideally process all survey steps. Two points of view must be considered on this matter: the developer and the user. For the developer, system coherence is dependent on the existence of a framework. This includes the consistent use of operating systems, foundation software, modules, function calls, programming objects, etc. With respect to the user's point of view, system coherence is achieved through the consistent use of graphical objects, concepts, variable names, data formats, codesets, etc. The coherence across systems directly helps users to shorten their learning curve. Furthermore, it usually reduces the maintenance and support cost for the whole suite of systems.

Accuracy: The accuracy of a system is the degree to which its outcome corresponds to what it was designed to process. Accuracy is ensured by a quality assurance process during the development phase that includes several phases of testing.

Timeliness: Timeliness can be seen from different angles. During the development phase, it refers to the delay between the day when the system is required and the day when it becomes available. During the practical use of the system, timeliness mainly refers to how fast it runs and whether or not it meets the user requirements in terms of speed. Like statistical products, the timeliness of systems influences their relevance.

4. THE SCOPE OF COMMON TOOLS

As mentioned above, when there is no commercial product available to run a specific statistical function and when no internal systems can be tweaked to fulfill the needs, the agency has no other choice than to plan for a new development initiative. Usually, work starts with statistical research projects. Statisticians or methodologists may spend years in trying and fine tuning methods. They go through several approaches and options, think about algorithms and implement prototypes that are tested on several datasets. At Statistics Canada, research projects can be split into two categories: the statistical research and the informatics research. Their goals are to improve the current program, either on the theoretical or the implementation side. They are funded based upon their potential benefits on the statistical program, with such benefits being reviewed every year against the increasing investment. Note that benefits are related to the quality of the end products and return on investments, whether directly through survey functions or indirectly through administrative and support functions. Since the benefits are weighted by the related volume of applications, a function that supports more applications will receive more funds. This principle drives the identification of common tools which are required by more than one project.

Let's first focus on survey functions. At Statistics Canada, several committees overlook software development initiatives and the required funds. We can classify these committees into two categories. The first mainly covers technical aspects. It includes (i) divisional and branch research committees, (ii) technical committees, (iii) informatics committees and (iv) function-specific committees like the Methodology Research and Development Committee, the Committee on Imputation Practices, the Committee on Disclosure Control, the Committee on Panel Surveys, the Data Analysis Committee, etc. The second category includes senior management committees like the Corporate Software Strategy Committee, the Methods and Standards Committee, the Advisory Committee on Statistical Methods, and other advisory committees. Together, all committees try to make efficient use of new methods and techniques through the development of common tools.

Sigman (1997) noted that a statistical agency must provide an answer to "*What are the statistical methods that should be implemented to be used across surveys*?" in order to prioritize the development. He was thinking about the generalization of systems, more specifically, he was justifying the scope of the Standardized Economic Processing System (StEPS) for the U.S. Census Bureau. In its generalization exercise 15 years ago, Statistics Canada also had to identify priorities. At that time, the main goal was to support the most common methods of sampling, collection, edit and imputation, and estimation that were being used. At the same time, it was recognized that some flexibility also had to be built into the functions. In this way, the duplication of effort across the customized systems could be reduced. This duplication had been occurring for many years and demanded a generalized solution.

Certain STC generalized systems now appear to be mature enough in terms of the functionality they offer. This has prompted managers to focus the current objectives on enhancement of the structure of the systems. The measures being pursued include the

development of completely independent modules for all systems, adding flexibility in terms of the computing platforms and database formats, enhancing the user interfaces, and further integrating the existing customized tools into the generalized systems.

The identification of common tools depends on the funding mechanism. This is because some tools may require investments in order to generate savings. Let's then present a few funding principles. STC funding decisions are made on a hierarchical basis, starting with Project Review Teams that collect proposals and make recommendations to Senior Management Teams – also called Syndicates – which in turn make recommendations to the Corporate Planning Committee (CPC). Every year, the CPC allocates funding and monitors the review of a Long Term Planning process and keeps decision records for every funded project.

In today's context, financial support is provided to initiatives that target efficiencies in any aspect of the survey program. Examples are the integration of surveys, the reduction of software diversity and platform diversity, the centralization of support teams, the maintenance of registers, the increased use of administrative sources as statistical products, etc. Efficiencies are not always measured in terms of dollars but also in terms of relation with data providers. Therefore, serious efforts are also put into reducing the respondent burden and into respecting their privacy. Software development proposals that target any of the above objectives will be put high on the bureau priority list.

5. SOFTWARE COMPONENTS

The developers have to plan for hardware and software components before starting to develop specific products. Such components should ideally fit their framework, should a framework be defined in their agency. Furthermore, the framework must be regularly reviewed to ensure that it conforms with the technical architecture and the emerging technology. Let's distinguish "architecture" from "framework" before going any further with technical aspects.

The information technology (IT) architecture defines the theoretical aspects, i.e. the IT design. It includes all system design decisions which are not related to logical performance. Its details span the full spectrum of technological considerations, like the choice of network protocols, the data communications agents, the inter-process communications, the operating systems, etc. The architecture is structured in software layers which ease the integration of various technologies. To be usable, the architecture must stay relatively stable.

A framework is a high-level roadmap for planning and defining information systems. As such, it is dependent on the IT architecture, while the architecture stays independent of any framework. The framework defines the practical environment within which the software components will run. As mentioned above, the framework should evolve with technologies. Systems being implemented within an agency are expected to adopt a common framework in order to achieve the benefits of consistency and cohesion. Statistics Canada (STC, 1996 and 2001) documented such an evolving framework.

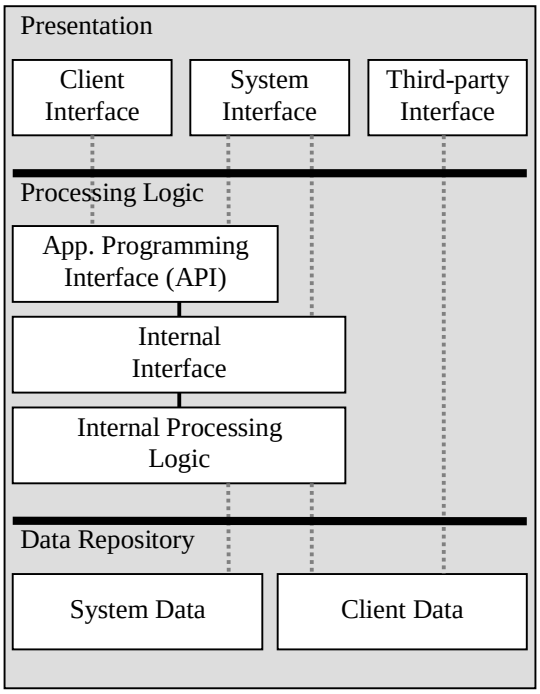
The developer must avoid the use of independent architectures, and even worse, they should not switch architectures between major releases of their products. From the user perspective, the lack of a framework increases the need for reformatting data between processing steps. In some cases it also increases the needs for costly third party software. At Statistics Canada, products used to be developed or redeveloped independently of each other. The result was a broad spectrum mix of technologies. These facts were the source of common complaints regarding the applicability and efficiency of products.

Approximately 15 years ago, a single data repository was encouraged for the development of common tools at Statistics Canada (STC, 1997). The repository of choice was SQL-based, and was available across three processing platforms in use within the Bureau: Micro, Unix and MVS. This was a significant step ahead in solving the disparate repository problem. Systems based on SQL hold the promise of vendor independence. In fact, the data processing community became so enamored with SQL that it made the language as powerful as portable. A whole new generation of software products has evolved to allow SQL to be used to access data on remote servers. The vision of performing all data manipulation within common systems using SQL is still as valid today as it initially was. In the past, STC systems that were developed using this technology were criticized for their inefficiency and for their requirement for the purchase and maintenance of Oracle, a relational database management system that can be seen as complex and expensive. This was especially true for systems mainly offering batch processing, with no real use of database features. In response to these criticisms, some systems have been redeveloped

based on the SAS system which offered the processing of both sequential data and relational data. Note that the capability to process relational databases is mandatory within a statistical agency in order to support on-line processes like data editing, data analysis and dissemination. Therefore, frameworks which allow the flow of data through sequential and relational structures are preferable.

A plan would start with the identification of a preferred technical architecture, which dictates how systems will be created. Therefore, the plan must address the choice of network protocols, data communication tools, inter-process communication tools including metadata, operating systems, programming languages, utility libraries, etc. Figure 1 shows the various elements of a technical architecture. These elements are grouped into three layers: the presentation layer, the processing logic layer and the data repository layer. Statistics Canada (STC, 1997) gives more details on the content of each layer.

Figure 1. Elements of a Technical Architecture



Presentation layer: This layer provides the user with the tools to set up and control the execution of the system. The first component of the presentation layer is the client interface. This component is developed and maintained by the user who requires the functionality of a specific product to be embedded within his own custom application. The second component is the system interface, i.e. the system built-in communication protocol. This interface may target a command line process or an interactive process in order to provide the expected access to the system's functionality. The last component is the third-party interface which offers the option of accessing the data through the use of vendor tools. Figure 1 clearly shows the direct access of data from the third-party interface, with no need for any component of the processing logic layer.

Processing logic layer: This layer includes the main processing of the system. Its first component is the application programming interface (API). This is mainly a set of stable modules or functions that can be embedded within a client system. The well documented API function calls are usable by both the client and system interfaces. The second component, the internal interface, provides an internal-level interface to the main processing module of the system. It ensures that changes to the internal logic remain concealed from the external view. Usually, the API reduces the needs for accessing the internal interface directly from the presentation layer. The last component is the internal processing logic, the heart of the system. It performs all the significant work of the system and is typically the largest and the most complex component. From the user perspective, the complexity of the processing logic is the motivation behind the development of a well-defined system interface and API.

Data repository layer: This layer stores information that is required by the system, either as input or output. The management of data is often achieved through a database management system (DBMS) like Oracle or a proprietary file system like SAS. The developer has to weigh pros and cons of DBMS vs. file systems when making a choice. While the first offers multi-user access along with data integrity, it is often slower and more expensive than the second. As shown in Figure 1, the data repository includes data that is used solely by the system. It is accessed only by the internal processing. The repository also includes client data and metadata, i.e. the data that belongs to the user but which is accessed and modified by the system.

The identification of the best interface technologies is far from easy. It obviously depends on the hardware infrastructure as much as it depends on the level of interest for a tight binding with specific commercial products. There are technologies available which allow execution across platforms. Unix emulators may help PC users, while Unix users can install Windows environments to run Windows based interfaces. The emulation feature represents a

solution to portability but requires the purchase and the maintenance of additional packages. The XML technology is promising with respect to data interchange between software components that don't necessarily run within the same environment. The power of .NET and JAVA technology must also be mentioned here. Since .NET supports multiple programming languages, it may offer a solution for the integration of systems. Given all this, the developer should consider foundation software that is portable by nature in order to reduce the need for extra packages, should a migration of platform become necessary. Standard C code provides the required portability but it is seen as difficult to code and to maintain. SAS products may be an alternative but their acquisition and renewal costs are an important factor. This may represent a risk for a statistical agency.

The objective of a statistical agency is to collect data from various sources and to make inferences from them. In fulfilling its mandate, the agency is very dependent of its data providers, most often survey respondents. From the agency point of view, current issues include the maintenance of social trust and confidence, and the respect of privacy (Fellegi, 2003). This issue has several implications, some political and some others more practical. Given the scope of this paper, we focus on one practical IT implication: the security of data. Statistics Canada (STC, 2001) requires that the agency have a coherent and consistent security environment. This environment must attain a standard that meets its obligations under its mandate, supports access to information and privacy principles, and reinforces public confidence in the way that confidential data are handled. To meet these requirements, system developers must assert that protected data is inaccessible from public channels. However, the approach requires careful design for applications such as electronic data reporting (EDR). Even within the agency, special security procedures must be developed to ensure security of data. Limited access privileges to data, functions or systems must be planned at the design phase.

6. INTEGRATION

In system development projects, there generally is a preferred approach which targets modular components. Many statistical agencies develop systems as small building blocks which can be integrated to a wider infrastructure. Blocks that correspond to embeddable components would most likely fit the API aspect of the desired infrastructure (see section 4.), and their integration with other survey steps would then become easier. Here, the integration refers to the flow of data across various survey steps. A well integrated suite of systems would allow the processing of data with minimum reformatting or pre/post processing. Survey managers realize that key elements of statistical programs are their data editing components. This is because (i) the editing is a complex process with online and batch activities, (ii) it interacts with most of the other survey steps, and (iii) it offers an important source of information for the improvement of survey programs over time. Therefore, a developer may want to focus on the edit process while developing a statistical infrastructure that allows information to be shared from all angles.

A few years ago, the United Kingdom Office for National Statistics (ONS) initiated a modernisation project which targets the delivery of a standard technical infrastructure, along with standard statistical tools and survey methods. ONS objectives address the issue of software diversity by focussing on a smaller range of standard software. The integration of an information management system and the development of a centralized data repository to hold all forms of data are other objectives of the project. In this context, the hope is to process all survey steps with minimum data reformatting or manual interventions. Tate (2003) explains that the implementation of metadata is the key element to achieve smooth processing of surveys, not only for each specific survey occasion but for series of regular surveys. For that reason, it is strongly suggested to plan for metadata as part of a centralized technical infrastructure. Metadata must be structured to provide information at various levels, from the description of variables with the sources of their content, the description of individual records, datasets, methods, classification, quality statements, up to the description of surveys, whether longitudinal or cross-sectional.

7. SOFTWARE MANAGEMENT

As reported in Statistics Canada (2001), the software management requirements usually consist of maintaining operational readiness of systems in production, including the ability to resume operation after unplanned outages and technical failures without loss of data integrity. It should also require the maintenance of a register of software assets to make visible those components that can be reused, to record the dependencies among components, and to

maintain a history of software investment. The registration of software would provide the basis for planning maintenance activities, for estimating maintenance costs, for assessing risks and impacts of software changes, for monitoring the compliance with official languages policies, for locating local documentation and for identifying internal resource persons or groups.

Statistical agencies should actively encourage the sharing of best practices and value experience and expertise equally. To ensure this, agencies should seriously consider forums where best practices would be promoted. At Statistics Canada, a software best practices group was put in place to improve the processes for developing and maintaining software applications. The group tries to promote high quality IT solutions as well as consistent development and maintenance activities. It follows a collaborative approach, engaging the participation of managers, developers and maintainers.

8. CONCLUDING REMARKS

This paper does not give any foolproof recipe to develop a set of statistical processing tools because, in fact, they are too dependent on the technical environment of statistical agencies. It rather describes principles that help in building expertise. There are several aspects of the expertise that were not really addressed, like the staffing issue (what should be the best mixture of programmer-analysts and methodologists for a software development project), the training aspects (recommended courses, assignments, term rotations, etc.), and the retention of staff.

Rather, the paper focuses on more technical aspects like the identification of common needs, the reduction of software diversity, the development and promotion of an IT architecture and framework, the effort in integrating systems, the promotion of good practices and the evaluation of resulting systems according to well defined quality dimensions.

ACKNOWLEDGMENTS

The author would like to thank his colleagues and the referees for their insightful comments.

REFERENCES

Bankier, M., Lachance, M. and Poirier, P. (1999). "A generic implementation of the New Imputation Methodology", *Proceedings of the Survey Research Methods Section*, American Statistical Association.

Fellegi, I. P. (2003). "Official Statistics – Pressures and Challenges", ISI President's Invited lecture.

Kozak, R., Poirier, C. and Kovar, J. (2001). "Generalized Survey Processing Systems: The Canadian Perspective", *Proceedings of the Survey Research Methods Section*, American Statistical Association.

Outrata, E. and Chinnappa, N. (1989). "General Survey Functions Design at Statistics Canada", Statistics Canada Technical Report.

Sigman, R. S. (1997). "How Should We Proceed to Develop Generalized Software for Survey Processing Operations Such as Editing, Imputation Estimation, Etc.?", U.S. Census Bureau Technical Report.

Statistics Canada (1996). "Information Technology Framework", Statistics Canada Technical Report.

Statistics Canada (1997). "Generalized System - Architectural Unification Study", Statistics Canada Technical Report.

Statistics Canada (1999). "Generalized Systems: Products and Services", Statistics Canada Technical Report.

Statistics Canada (2001). "Information Technology Framework", Statistics Canada Technical Report.

Statistics Canada (2003). "Statistics Canada Quality Guidelines – Fourth Edition", Statistics Canada Catalogue no. 12-539-XIE.

Tate, P. (2003). "The Data Editing Process Within the New Statistical Infrastructure of the Office for National Statistics", Working paper no. 10, Proceedings of the UN/ECE Work Session on Statistical Data Editing.

Turner, M. (1994). "General Survey Processing Software: An Architectural Model", Statistics Canada Technical Report.