Distr.
GENERAL

WP.4
25 April 2012

ENGLISH ONLY

**UNITED NATIONS ECONOMIC COMMISSION FOR EUROPE (UNECE)**
**CONFERENCE OF EUROPEAN STATISTICIANS**

**EUROPEAN COMMISSION**
**STATISTICAL OFFICE OF THE EUROPEAN UNION (EUROSTAT)**

**ORGANISATION FOR ECONOMIC COOPERATION AND DEVELOPMENT (OECD)**
**STATISTICS DIRECTORATE**

**Meeting on the Management of Statistical Information Systems (MSIS 2012)**
(Washington, DC, 21-23 May 2012)

Topic (i): Architecture

# Value of Systems Architecture for Statistical Organizations

**Supporting Paper**

Prepared by Juan Muñoz López, INEGI, México

## I. Introduction

1. Systems architectures are instruments that in some cases are undervalued by organizations which are more worried about how to solve specific problems in short time than having strong and flexible systems with an expected long lifecycle. These organizations are forgetting that a well-designed system is one that is capable to be adapted to several circumstances with little or no modifications, condition that is necessary to streamline the statistics business process.

2. Systems architectures can be extended from specific solutions to a whole computational environment. Even though the design of the last ones is more complex, the benefits in interoperability, reusability and governability worth the effort.

3. At the end of this paper, some cases of systems with flexible/open architectures implemented by the National Statistics and Geography Office of Mexico will be reviewed.

## II. Relation between Software Architectures and Statistical Processes

### A. System Architectures

4. Software architecture describes the structure of a system that is intensively based on software; it describes the structure or structures of the system, which comprise software elements, the externally visible properties of those elements, and the relationships among them (Bass, Clements, & Kazman, 2003). A software architecture document can be seen as an instrument which serves for multiple purposes like inception, analysis, planning and supervision of the system and its development; and also it's useful for communication and coordination between all the people that is directly or indirectly interested in the system.

5. Rozanski & Woods (2005) says that architecture of a system defines four different aspects:

(a) Its static structure, tell us how elements of a system have been designed, and how they are arranged.
(b) Its dynamic structure, describes how elements will be working when the system is running.
(c) Its externally visible behaviour, defines how external users and other applications in the computational environment will be interacting with the system.
(d) Its quality properties, which are non-functional characteristics of the system, but they serve to qualify how it operates, interacts and gives results.

6. Intrinsically, all systems have an architecture, no matters if it is documented or not, if it was planned or not; and this will determine various quality and functional aspects like its capability to evolve and to integrate new capacities, its adaptability to environmental changes, its durability, and so on. We can opt for start a “paved road” with a good architectural design before making the system, or just let us go to anywhere where the "organic code" could take us through its disordered growing.

7. A short-sighted architectural design will bring solutions that only will be useful to solve immediate problems within a very limited scope. In our area there are multiple “stove pipe systems” which prove this statement. But, good software architectures will help us to get more robust systems that can be adapted to different situations.

8. Architect’s vision can take us from narrow solutions just focused on solving specific needs, to flexible systems capable of managing different situations from several domains for different areas in the organization; his broad perception of the technological background will result in efficient components that will interoperate inside and outside the system, or in incomplete and isolated components incapable of being integrated with its environment; his sharpened brain can inspire solid and clear systems or weak and complex computational nightmares.

9. Well-designed systems represent less work, because the effort needed to increase its functionalities and satisfy new requirements decreases as far as the system lowers its complexity and it can be better understood by the team of developers. And of course, it will be cheaper to maintain because there will be a smaller probability for the apparition of new bugs, which most of the time will be easily corrected.

10. Designing a system is a collaborative work; the role of system architect has more from integrator than from director. Indeed in many organizations this role is not assigned just to one person, but to a multidisciplinary team in charge of doing this job.

11. System architect must envision a solution incorporating many points of view that represent different interest from all people directly or indirectly involved with it. These ‘stakeholders’ can characterize a broad gamma of roles, like:

(a) Sponsors. Will be paying for the system, they are interested in the final objectives, the impact in the organization, the budget and time to have the solution, etc.
(b) Conceptual specialists. They are experts in the subject matter domain addressed by the system; they know methodologies and technics to solve the problems and how results must be represented and interpreted.
(c) Final users. Are people who will interact with the system or just receiving the results; they can tell us how to make the system more practical. In some cases there are a lot of different profiles of users that we need to contemplate, and in some cases we need to consider usability issues to make it accessible to handicapped people.

(d) Software architects. Responsible for designing the system; they need to integrate all the points of view to make a solid document that will represent a solution to satisfy agreed needs of all ‘stakeholders’.
(e) Project managers. They need to manage the process to develop a solution that their team can deliver on time, under the estimated budget and covering all planned goals.
(f) Software developers. They make the code of the system. They need to clearly understand the scope of the system and be sure that they can traduce all the requirements to functionalities and characteristics of the programs.
(g) Systems quality team. The staff of this team need to find in the software architecture a kind of checklist to guide the revision process in which they will ensure that system satisfies all functional and no functional requirements without visible defects
(h) Technical support staff. They need to know the technical requirements to incorporate and to operate new solution and decide if the new system, resources and technologies that will be supporting it are adequate for the IT environment of the institution.
(i) Network managers. Responsible of electronic communications in the organization, they need to be sure that the network will not be degraded with the new system. In some cases they will determine if it is necessary to expand or to add new features to the network.
(j) Security Information Staff. They deal with a wide variety of threats that potentially put organization in risk. They need to be sure that the new system will not create security threats.
(k) Database Administrators. They need to be sure that the database logical and physical designs are correct, useful and complaint with institutional standards to maintain information query-able, congruent, confidential and dependable.
(l) Outsourcing providers. They are external sources of different kinds of resources (goods and services) required to build and deploy the system. They can provide very valuable information about potentials and limitations of current technologies.
(m) Auditors. They will ensure that system will comply with standards and regulations and that the process to implement it runs under the applicable legislation.

12. Documenting software architectures isn’t a trivial work. We must represent stakeholders’ concerns in a language clearly understood for all them. Most of the time we use many diagrams in some technical language like UML to represent different specialized aspects, then they are complemented with informal drawings to make high level representations of some relevant parts of the systems for the less skilled people. Those diagrams are known as “views” and they are clustered in groups called “points of view”.

13. Well documented software architectures are the basis to construct efficient IT environments in which we can share resources among different solutions and combine information and functionalities of them to get a synergy effect to create more powerful systems which can interoperate and complement each other. When we have implemented a system based in a documented architecture we can evaluate decisions we take about implementing it and changes that we made to the original design, and extract the knowledge to improve the development process. If we decide that the architecture was good enough we can synthetize its main characteristics and document a reference architecture that we can use as to support the design of new systems.

14. As a communication instrument, reference architecture can help to define the common elements of solutions which can be used by different organizations to implement its own solutions; and in a better approach, they can be used to define compatible modules that can be exchanged and shared among institutions to build customized systems incorporated to different environments.

## B. Industrialization of Statistics

15. Producers of official statistics have been stressed by a complex environment where as a result of financial crisis their budgets have been reduced, although they need to generate a huge number of products to satisfy requirements from society. Users expect to get statistics that they can easily analyse with modern IT tools to make informed decisions. Right now, emerging technologies raise expectations of society for having quick snapshots of real world measures.

16. To help statistical organizations to maintain relevance of official statistics, the High Level Group for Strategic Developments in Business Architecture in Statistics (HLG-BAS) have defined a vision composed of two directions (Statistics Netherlands, 2011):

(a) Improvement of Statistical outputs with better products and services created from a global perspective to maintain official statistics relevant and tuned with current user´s needs.
(b) Creation of new and better production methods to improve efficiency and effectiveness in the delivery of products at a minimal cost, with greater flexibility and in cooperation between institutions.

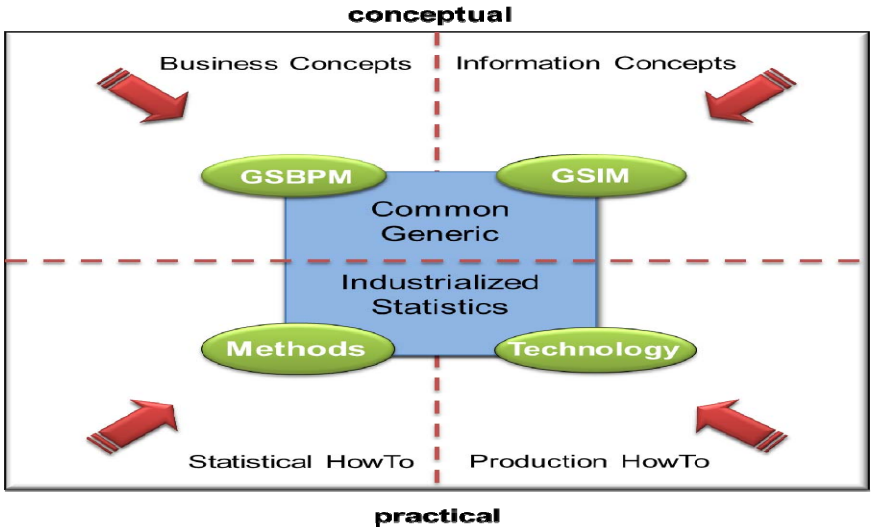
17. Industrialization of statistics implies that expensive statistical products that are being made case by case, in an artisanal way, are streamlined and developed conforming to standardized processes in a way the cost is considerably reduced. Those processes are designed in such way that they can be easily customized to produce statistics from different subject-matter domains.

18. The HLG-BAS says that industrialization of statistics aims to lower cost of delivering products to satisfy users' needs by mean of the standardization of processes and products. In this context, lowering costs while increasing effectiveness is conceived dividing the process in four phases:

(a) Product design; design of products constrained by manufacturability.
(b) Process design; design of modular production process independent of the subject-matter domain in which methods and process can be exchanged between institutions.
(c) Production; production executed by machines as close to real time as possible.
(d) Analysis; publishing and research made by statistical subject-matter experts from the analysis of outputs and intermediate results assisted by advanced tools with the least possible human intervention.

19. Those four key phases can be considered as required tasks to create a new process in which production of statistics can be taken to an industrial level. By industrial level we´re referring to methods of production that are standardized and optimized aiming to get products for users' needs, in short time with low costs.

**Figure 1; Industrializing Statistics (Statistics Netherlands, 2011)**




20. In the vision of the HLG-BAS, the new statistical process must complaint with four constraints (Statistics Netherlands, 2011):

(a) The Generic Statistical Business Process Model (GSBPM). A flexible tool to describe and define the set of business processes needed to produce official statistics. This model is also used for harmonizing statistical computing infrastructures, facilitating the sharing of software components, in the Statistical Data and Metadata eXchange (SDMX) User Guide for explaining the use of SDMX in a statistical organization, and providing a framework for process quality assessment and improvement (UNECE, 2009).

(b) The Generic Statistical Information Model (GSIM). A reference framework of information objects, which enables generic descriptions of data and metadata definition, management, and use throughout the statistical production process (GSIM draft version).

(c) Methods. Practical implementation of the models in the conceptual area.

(d) Technology. Set of standard solutions to produce statistics.

21. A generic statistical industrialized production process like this one will require strong support of information systems designed with a new approach that will need to address the design of modular and extensible systems capable of incorporate, eliminate or replace functionalities as needed by different subject-matter domains in short time and with a low cost.

22. In a context like this, where industrialized processes are going to guide the development of systems, an approach of software product lines for the design of architectures for statistical information systems is a logical way to get reusable applications that maximize commonalities among software programs that could be considered part of a family of products, while providing sufficient variability for each member of the family to address specific needs in a determined domain.

23. A software product line consists of a product-line architecture and a set of reusable components that are designed for incorporation into the product-line architecture (Bosch, 2000). As we can see, this approach is based in reusable architectural designs and in component-based software.

24. A software component is a unit of composition with explicitly specified provided, required and configuration interfaces and quality attributes (Bosch, 2000). In real life a component can be a small piece of software or a whole system. We can reuse components at different levels: in different

modules of one version of a system, over subsequent versions of a system, over different versions of various related systems, over different versions of various related systems in different organizations, and so on. To get necessary variation of those components, we design them in such way that they can be configured, specialized or easily replaced by providing standardized interfaces.

25. Under the products line approach, we can develop systems that are explicitly created to be reused for different statistical projects and which can be shared among organizations. Also, creation of components that can be specialized or exchanged as needed to support specific issues is another benefit from this way of conceiving software design.

26. Jacobson (Jacobson, Martin, & Jonsson, 1997) defines a process of three stages to create rusable software in a product lines approach:

(a) Application family engineering. Design an architecture which can cover all the products in the product line and includes features that can be shared between products
(b) Component system engineering. Design of common components that have been identified in the architecture, we need to describe their functionality, interfaces and variability.
(c) Application system engineering. Design of a specific system which is part of the family of products. In this stage we adapt the generic architecture of the product-line to fit our needs by adding or replacing components and relations between them, developing specific extensions, configuring components and designing the specific code that the system will need.

27. As we can see, this process is guided in a top-down way, we need to go from the most general to the most particular. This process has been taken by Bosh (Bosch, 2000) to propone a six steep method for the design of a product line software architecture:

(a) Business case analysis. First we must review if the additional effort to develop a product line will represent benefits. Only architectures of systems that will be widely adopted or that will have different implementations are candidates to follow this approach.
(b) Scoping. This activity determines the products and the product features that are included in the product line. We need to choose only products that are related accordingly to an established plan, features must be selected from the main requirements that the set of products of the family must fulfill.
(c) Product and feature planning. In this step we are going to plan versions and features that we'll be developed in the future.
(d) Design of the product-line architecture. This is the main step of the process. Besides activities to design architectures for specific systems, we must design a valid architecture for various products, including common and specific features. The design must prevent potential conflicts, required variability, evolution of the systems, interchangeability and optionality of features and components.
(e) Component requirement specification. This activity is concerned with specifying the requirements for each component described for the architecture. The requirements specification defines the interfaces, the functionality, the quality attributes and the variability that each component should support.
(f) Verification. Before starting software development, it's important to verify that the product-line architecture supports the requirements specified by the stakeholders. After this step we're going to design specific systems based in the product-line architecture.

28. Under this approach we're not envisioning just one system, we are designing a group of systems that will be part of an integrated platform in which systems will easily interoperate and share resources. With the product-line architecture serving as a coordination instrument, different organization can participate in the creation of different components that will be used in several systems.

## C. Getting Benefits from Software Architectures

29. Documenting software architectures is an activity that is undervalued by a great number of organizations because they don't perceive an immediate value in them. Maybe because architectures are not executable products that they can deliver to users. But we have seen that software architectures can help us to build better systems, with less effort and investing fewer resources. The benefits that we get from architecture depend in great part on how well it has been designed.

30. Benefits from architecture are also related to the scope of what we are designing:

(a) Designing a single system, will help us to get a solution for a problem with an extended lifetime, capable of evolving and to be adapted to new requirements with less effort and cost

(b) Designing a family of systems, will bring systems to satisfy different requirements of a subject-matter domain. Those systems will be capable of interoperate and share information among them. A family of systems will also have interchangeable and customizable components that can be developed by different organizations.

(c) Designing a platform of systems, will be the basis of an information system's environment that will serve to support the whole statistical process and to incorporate solutions that will help to support interconnections with the systems of other organizations and to bring information channels to users that will be able to connect its own analysis tools. So users will have Statistics as a platform.

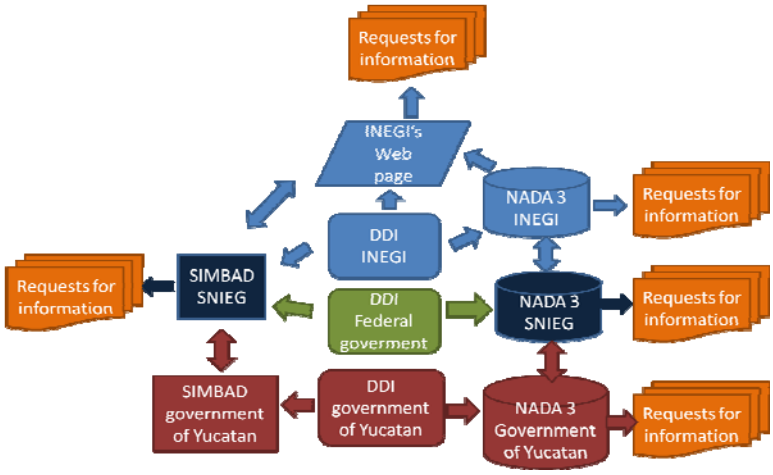
(d) Designing enterprise architectures will produce an environment of people, information, technology and processes that will be totally interconnected in a standardized environment focused on the evolution of the organization where the use of resources is optimized.

# III. Study Cases

31. We have a long road in front to reach a maturity stage in which all systems created by INEGI can be connected into just one platform to conform a whole integrated platform, but various systems that have been implemented can be useful to depict the potential benefits of following this path.

## A. SIMBAD (State and Municipal Databases System)

**Figure 2: Network of SIMBAD instances**

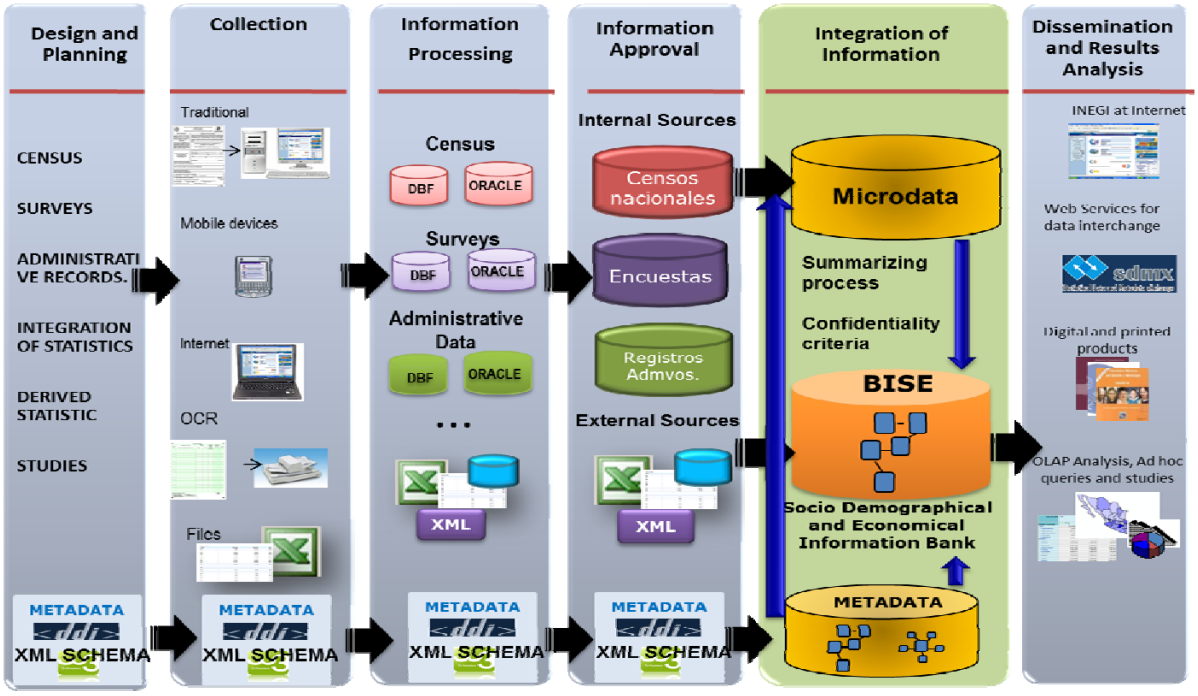


32. This is one of the oldest systems at INEGI; it's a system that was conceived to combine information of databases from different statistical projects making data comparison easier, this solution provides tools to generate indicators, make some OLAP analysis and to generate thematic maps.

33. As it was conceived to interconnect different databases, its architecture is appropriated to create networks of instances of the system distributed among different institutions sharing information and functionalities. At this time developers of this system are working on making the necessary adjustments to concrete this network.

## B. BISE (Socio Demographical and Economical Information Bank)

**Figure 3: BISE seen as a system for data integration of internal and external information sources**




34. BISE is an evolution of the Bank of Economical Information (BIE) that is running from some years ago at INEGI. It's a system that has been created to integrate socio-demographic and economic indicators from internal (like the statistical data-warehouse) and external sources (mainly systems from other federal government organizations)

35. The system has been conceived to be a data integrator and a data provider for other systems which can connect to it by using web services.

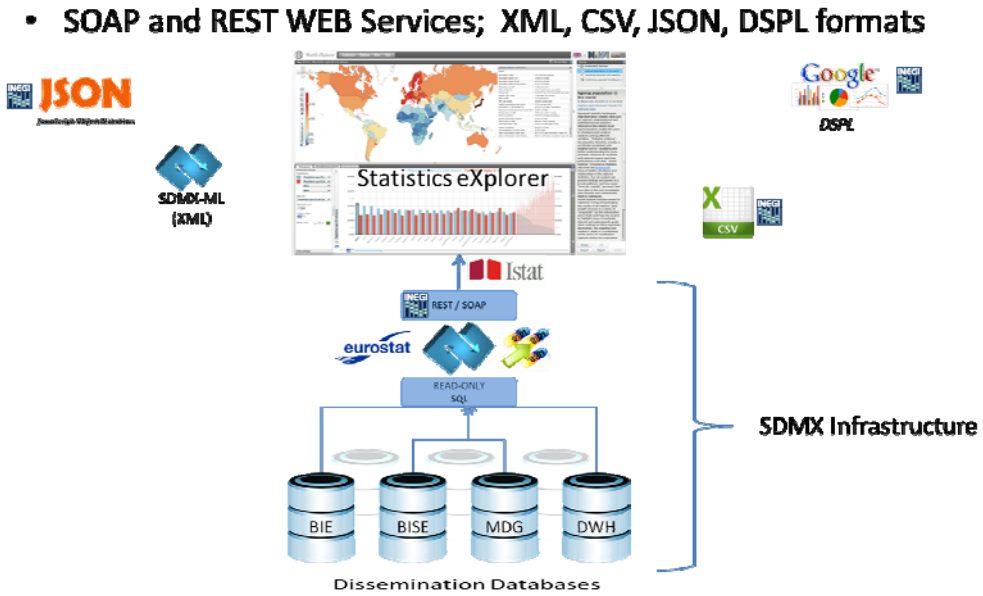
## C. SDMX Infrastructure of INEGI

36. Infrastructure for transformation of information to SDMX can show us how we take advantage of using open architectures that can be combined to develop powerful solutions. In this case the base of the INEGI's SDMX infrastructure is the Reference Infrastructure created by Eurostat which can connect our dissemination databases and create SDMX data flows.

37. We have created extensions to support REST web services and to deliver SDMX data flows in different formats like: CSV that can be used by a wide range of applications like Excel; DSPL that can be used by the Google Public Data Explorer; JSON and JSONP that can be used directly by systems of other

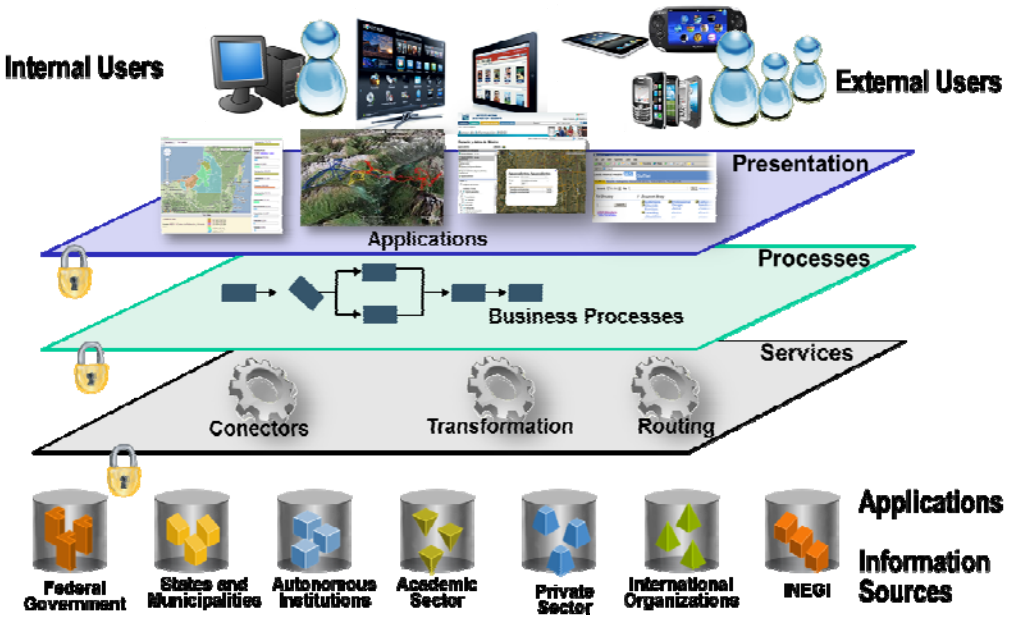
institutions and; Chart to be used by graphics. We are now working in a specific XML format that can be used by means of a solution created by Istat (National Institute of Statistics of Italy) to feed information in a visualization and analysis tool calledStatistics eXplorer.

**Figure 4; SDMX Infrastructure of INEGI showing how resources from different institutions can be mixed to create powerful solutions**




## D. SOA Planned Architecture for Mexican National Information Network

**Figure 5; High Level Detail of the National Information Network IT Architecture**



38. A long term goal for INEGI is to integrate a National System that will be connecting systems from different institutions in a big network of geographic and statistical information. This network will be possible by generating an open platform capable of integrate different processes and technologies.

## III. Conclusions

39. Although in many cases software architectures are undervalued, they are powerful coordination and communication instruments. They are recommended for statistical organizations who want to improve scope and quality of their systems. We must remember that software architectures are critical to define if a system can support certain functionalities and incorporate desired quality characteristics.

40. To adopt the right methodologies for the design of systems architectures could help to improve systems development processes, to reduce efforts and to lower costs. This is especially valid in a statistics industrialization context. Methodologies for the design of architectures for product lines could be a valid approach to create systems with desirable characteristics like reusability and interchangeability of its components, interoperability, flexibility and adaptability.

41. Designing architectures with a wide vision helps to build complete IT environments that bring benefits to organizations and set the basis for the cooperation of institutions for development of common solutions.

42. Satisfying new user's requirements like open access to data, incorporation of powerful analytical tools, integration of data from different domains, dissemination of information to new devices, etc., requires of robust systems for which architectures must include services that brings open, direct and flexible ways of connection.

## IV. References

Bass, L., Clements, P., & Kazman, R. (2003). *Software Architecture in Practice. Second edition.* Boston, United States of America: Addison Wesley.

Bosch, J. (2000). *Design & use of software architectures. Addopting and evolving a product-line approach.* London, Great Britain: ACM Press/Addison Wesley.

Jacobson, I., Martin, G., & Jonsson, P. (1997). *Software Reuse: Architecture, Process and Organization for Business Success.* New York, United States: ACM Press/Addison-Wesley.

Muñoz López, J. (September de 2008). Methodology for the Design of Reference Architecture Models which Integrate Legacy Systems. Aguascalientes, México: UAA.

Rozanski, N., & Woods, E. (2005). *Software Systems Architecture.* New Jersey, United States of America: Addison Wesley.

Statistics Netherlands. (2011). Strategic vision of the High-Level group for strategic developments in business architecture in statistics. *Conference of European Statisticians* (págs. 1-9). Geneva: UNECE.

UNECE. (2009). Generic Statistics Business Process Model Version 4.0. *Joint UNECE/Eurostat/OECD Work Session on Statistical Metadata METIS* (págs. 1-28). Geneva: UNECE.