Bédard, Y., 1989, Information engineering for the development of spatial information systems: A research agenda, 27th Annual Conference URISA,Vol. IV, p. 43-53, 6-10 août, Boston, Massachusetts, United-States

Information Engineering for the Development of Spatial Information Systems: a Research Agenda

Y. Bédard

Proceedings of the 27th Annual Conference of the Urban and Regional Information Systems Association
Vol. IV, pp. 43-53.

Boston, August 6-10, 1989

Dr. Yvan Bédard, Director

Laboratoire de systèmes d'information à référence spatiale

Centre de Géomatique

Dép. Sciences Géodésiques et Télédétection

Université Laval

Ste-Foy, Qc, Canada, G1K 7P4

Tél.: (418) 656-3694

FAX.: (418) 656-3177

BITNET: 1130025 @LAVALVX1

INFORMATION ENGINEERING FOR THE DEVELOPMENT OF SPATIAL INFORMATION SYSTEMS: A RESEARCH AGENDA

ABSTRACT: Implementing large Spatial Information Systems (SIS) brings very important problems like figuring out what users really want, adapting the technology to the organizational needs, and implementing new operational/administrative procedures. Actually, the field called Information Engineering builds and uses structured information system design methodologies, as well as specific tools (e.g. data models, data flow diagrams) and techniques (e.g. prototyping) to solve these problems for non-spatial systems. Despite claims of benefit in productivity and quality results, it is only recently that Information Engineering has gained large acceptance. This new popularity is closely related to the growing complexity of information systems and the arrival of a new generation of software tools called CASE (Computer-Assisted Systems Engineering). This field, Information Engineering, can be of great benefit to SIS, especially after adaptation. This paper presents Information Engineering fundamental concepts and research issues involved in extending traditional system design methodologies and tools for Spatial Information Systems.

INTRODUCTION

Developing Spatial Information Systems (SIS) involves much more than technology. It also involves the coordination of human resources, procedures (manual and automated) and data (of various forms in various formats) within a certain budget and an overall organization. Often, such development also requires one to consider other organizations because several kinds of spatial data come from external systems (e.g. cadastre, assessment, census, topography); this is mostly due to high data collection costs and the institutional nature of some data.

The new field called Information Engineering already addresses such complexities; for example, it addresses questions about the system design methodology to adopt for the development of an information system, the analysis tools to use, the software tools to use, the techniques to choose, and the final results to expect. However, for the most part, Information Engineering concepts are applied to non-spatial information systems; only very few organizations are starting to use Information Engineering for SIS.

This paper begins with a presentation of Information Engineering: its philosophy, its content and its goals. Then, the applicability of Information Engineering for the development of Spatial Information Systems is discussed. Actual limitations of Information Engineering for the development of SIS with regard to spatial referencing are also presented. Finally, R&D suggestions to improve Information Engineering for SIS are put forward.

INFORMATION ENGINEERING

In the late 70s and early 80s, SIS developers were navigating in a new field and most of them created their own development "methodology" based on intuition, experience and computer techniques. Their backgrounds were mostly in earth-related disciplines (e.g. surveying, mapping, civil engineering, natural resources, geography, assessment) and the emphasis of their "methodologies" was more oriented toward mapping than database management. However, the concept called "the communication paradigm of maps", in addition to the advancement of DBMS (Database Management Systems) and the growing complexity of SIS helped these developers to approach maps from a database perspective in the mid-80s.

In the meantime, non-spatial information systems (e.g. Banks, Hospitals, Schools) were developed with a different approach: a database oriented point of view. People involved in these systems were mostly computer scientists, managers and other database specialists. The developed and used tools (e.g. Data Flow Diagrams, data modeling, fourth generation languages) and methodologies (e.g. Gane and Sarson, Merise, Warnier-Orr) which gained large acceptance. Some of these methodologies were built on solid theoretical backgrounds.

These tools and methodologies proved useful especially for large systems where an organized approach was needed for development. A recent survey in France (9) showed that productivity gains in companies using formal methodologies for an average of 3.5 years was between 20% and 30%. Although "formal methodology" was not clearly defined in this survey, this reflects the general acceptance that the inclusion of well-defined procedures in the development of systems has its benefits.

It is from this latter group that the new discipline called Information Engineering emerged. This discipline integrates *tools*, *techniques* and *methodologies* for the development of Information Systems. Its goal is to create a *structured framework for an entire organisation* into which individual information systems, or sub-systems, will fit (11). Such an organizational framework is based on computerized data models, process models and enterprise models. It provides a general foundation on which to build the organization's Information System as a whole. The resulting system reaps greater benefits than the sum of individually built departmental systems.

Information Engineering philosophy for information systems development

Information Engineering *focuses on the organisation's data structure and the information derived from it*. It is considered implicit that the data types (entity types, attributes, relationships) change less frequently (although the values of the database continuously change) than the procedures and equipments (hardware, software) used in the organization.

It is upon this data structure that the other components of the information system are built. The data structure for the *entire organisation* is used, not that of any one sub-system. Such an approach creates a synergistic information framework for the overall organisation rather than structuring independent departmental systems within the organization. This global view is necessary in make all data compatible and to minimize redundancies and inconsistencies among the domain-specific databases of the organization (e.g. engineering, assessment, and planning departments of a municipality). Such a global view also encourages agreements over definitions of entity types and attributes across the organization. It increases the benefits related to the collected data as they can be more easily exchanged among databases and used for a larger variety of purposes; such benefits are especially important for spatial data, which are very expensive to collect and integrate.

Nevertheless, this approach does not require to develop the information system for the entire organisation all at once. It rather allows for the coordination of the information throughout the enterprise while domain-specific information systems can be implemented as separate modules according to the strategy of the organization. The implementation of these modules can hence be spread over time.

With this approach, it becomes possible to *use and integrate methodologies, techniques and tools* for different sub-systems of the organization as long as they fit within the enterprise-wide models. For example, we can use prototyping for a small but highly uncertain sub-system and a traditional system design methodology for another sub-system which is large but stable). Such flexibility allows for the selection of more appropriate and effective ways of developing sub-systems while considering the overall system.

An additional aspect of Information Engineering is the stress on *formalizing the system*; its goal, steps, costs, benefits, definitions, codifications, etc. are all well documented. This makes the system less dependant upon specific persons and facilitates its maintenance.

Information Engineering also strongly encourages a *continuous participation of the users in the development process*. A good way to achieve this is to use prototyping techniques and tools, fourth generation languages, and integrated CASE tools covering the entire development cycle.

Information Engineering is therefore a systemic approach (which should be intuitive for the development of a "system'") especially suited to the development of complex systems, as is often the case for SIS. Such an approach allows developers to work at different levels of abstraction at different periods: the most general level covers the enterprise while the most detailed level covers each project within each domain of application. As previously mentioned, Information Engineering stresses the importance of having, right from the start, a complete picture of the enterprise at the most general level, while detailed levels can be built as each sub-system is developed. This approach uses the "divide-and-conquer" idea where a complex system is divided into inter-related smaller but more comprehensible sub-systems. Furthermore, Information Engineering provides flexibility in the choice of tools, techniques and methodologies used.

<u>Information Engineering content</u>

Information Engineering encompasses system design methodologies and the most recent developments in prototyping techniques and CASE tools (Computer-Assisted Systems Engineering). However, the novelty of Information Engineering does not reside in the novelty of the individual tools, techniques and methodologies used but rather in their full *integration and computerization*.

a) System Design Methodologies:

“Methodologies are the sets of rules, methods, and procedures that cover one or more stages of the systems (or software) development life cycle and is used by systems analysts and designers” (13). There is no consensus on the number of stages of this life cycle nor on their exact definition, but it is common to define the system life cycle as containing five stages: feasibility study, analysis, design, implementation, and maintenance. Some methodologies show a larger view and begin with a Director or Strategic Plan for the overall organization. Examples of such system design methodologies are AXIAL, Gane and Sarson, Martin’s, Merise, NIAM, DMR Productivity, P +, Yourdon’s, Jackson’s, and Warnier-Orr. They all share common characteristics as well as offering different elements.

The early system design methodologies followed the development life cycle in a linear manner. Such methodologies, now called traditional methodologies, fully predetermined the system on paper before the first line of program code was written. Such a process proved useful but problems meeting user requirements were soon encountered and the results led to some disappointments.

This communication gap problem between developers and users has now been solved in several methodologies by continuously involving the users and by adding prototyping techniques. Depending upon the way prototyping is introduced in a methodology, we call this methodology mixed or evolutionary (3).

b) Prototyping techniques:

Prototyping techniques are used to facilitate the communication between developers and users. Burns and Dennis (7) stated that the “users can point to features they don’t like about an existing system (or indicate when a feature is missing) more easily than they can describe what they think they would like in an imaginary system”. This is especially true for SIS where few users imagine all potential capabilities of GIS tools (e.g. spatial analysis).

In this context, prototyping becomes a development strategy which can be integrated into a formal methodology. This provides early removal of uncertainties related to determining users’ requirements. As a rule, we can say the higher the uncertainty related to the development of a SIS, the more we need to use prototyping (7).

Prototyping implies the building of “an early version of a system that exhibits the essential features of the later operational system” (1). The developer quickly builds a prototype and shows to the users what the system will look like and how it will function. It is an iterative process where the users comment on each version of the system up to a version which satisfies their needs. This allows the developer to explore users’ needs further than only on paper.

The resulting system or prototype can be discarded once the developer has achieved its goal (this is called *fast prototyping*) or it can continue to evolve to become the final system (this is called *evolutive prototyping*).

Prototyping is achieved with software tools such as DBMS Fourth Generation Languages (4GL) and CASE tools, including prototyping utilities (e.g. screen generators, report generators, menu builders). However, for the cartographic aspect of SIS, such tools are rare if they exist at all. This explains why pilot projects take so much time: no tools are available for fast SIS application prototyping.

c) CASE tools

One of the major problems with information system design methodologies is the amount of work required to develop structured graphs, models, dictionaries, etc. A lot of time is required to draw and edit all the diagrams created while developing the system. “Mark

Tilley commented at CASExpo that engineers used to spend about 10 percent of their time thinking about system flowcharts. The other 90 percent would be spent actually drawing them." (13).

Several developers abandoned system design methodologies for this reason (4). However, now that CASE tools such as IEW, Excelerator, and Consoi are available, the overload problem is becoming a vestige of the past.

CASE software allows for the automation of several steps in the information systems development process. It is "the automation of step-by-step methodologies for software and systems development from step-one planning to ongoing maintenance; it is designed to automate the drudgery of development and free the developer to solve problems" (12). Thus, a CASE tool is a piece of software which allows the developer to automate parts or all of the development stages. Several CASE make use of graphics and easy user-interfaces. One could say that they are tools for computerizing the computerization process.

The term "CASE tool" has been used for two different but interrelated purposes. First, it has been used as an acronym for Computer Assisted *Software* Engineering, focusing on the tools used in the development of computer programs (e.g. code generator, documentation generator). Now, CASE is increasingly used as an acronym for Computer-Assisted *Systems* Engineering, describing tools needed for the development of an information system (including analysis and design tools such as data flow diagrammers, data modelers, and organization charters).

The first CASE tools duplicated what we did by hand (e.g. data models, data flow diagrams, data dictionaries), each task of the development process being done by a stand-alone program (e.g. diagramming tools, screen and report painters, code generators, documentation generators). Most CASE tools automating one task did not work with tools automating the preceding or next task. However, new and future CASE tools will integrate these different tasks so that the output of the first one will serve directly as an input of the second one. Ideally, CASE tools should cover the entire development process.

CASE software is perceived as one of the most promising fields of computer science. "CASE tools are breathing new life into structured methodologies by providing automated graphics facilities for producing charts and diagrams, screen and report painters, data dictionaries, extensive reporting facilities, analysis and checking tools, documentation generators, and code generators (or links to them)" (12).

Information Engineering goals

One of the aims of Information Engineering is *to computerize as much as possible the development of Information Systems*. This objective requires the use of computer tools such as screen painters, fourth generation DBMS languages, prototyping tools, code generators, and other CASE tools either specialized in or integrating some combination of data modeling, data flow diagramming, the creation of data dictionaries, etc. As a result, less time should be spent on programming, documenting, and other technical tasks, leaving more time for the conception of the system and likely leading to better results.

Such a level of automation should also result in easier maintenance during the evolution of the system. "An objective of information engineering is to create systems that can be modified quickly so that changed procedures can be implemented at once when they are required" (11).

The computerization objective also requires the building and use of structured tools, methodologies and techniques whose results are amenable to computer implementation. This formality requires a rigorous, methodical approach which is closer to an engineering-like approach than the traditional, mostly intuitive approach.

Thus, it appears that Information Engineering's overall goals are to allow one to accelerate the development of systems, to break the barriers between users, designers, and programmers, to maintain better control over the development and evolution of the system, and to formalize the development of the system by better documenting its many components and procedures with the appropriate analysis tools.

APPLICABILITY OF INFORMATION ENGINEERING FOR THE DEVELOPMENT OF SPATIAL INFORMATION SYSTEMS

Building a SIS requires important investments and developers must always be very careful about the all too frequent "computerizing the actual problems" syndrome. We should always remember Humphries' (10) sentence about LIS/GIS development: "having visited about 800 "systems" in about 30 countries, my synopsis is that the vast majority of such "systems" are merely automating the inefficiencies of the past... The difficulties are not attributable to our understanding (or lack of understanding) of the technology or its capabilities but the immaturity of our understanding of the necessary infrastructure to manage such a system".

The Information Engineering approach incites developers to solve the problems *before* final computerization; the sooner the problems are solved, the less expensive it is to solve them. This approach also makes it easier to resist over-enthusiasm and to take enough time for strategic planning. Information Engineering forces everyone to focus on the enterprise's needs and the coordination of its information resources; this is important in the current era of excitement over new GIS technologies.

"Strategic data planning, if done efficiently with a proven methodology, does not cost much when compared to the hidden costs of chaotic data. It is largely a one-time expense, with only a minor cost for keeping it up-to-date.... The maintenance caused by absence of planning is enormously more expensive than the planning would have been" (11).

Since a Spatial Information System is first of all an Information System, Information Engineering concepts could serve as the basis for the development of SIS. Selecting and using the right methodologies, techniques and tools to develop a SIS not only can be done with an Information Engineering approach, but can lead to immediate benefits. Campbell, who has been working on the development of a SIS for EMR Canada Legal Surveys Division, mentioned (8) that "to support the integrated design of spatial information systems, structured system design principles and methodologies offer considerable benefits".

LIMITATIONS OF INFORMATION ENGINEERING FOR THE DEVELOPMENT OF SPATIAL INFORMATION SYSTEMS: RESEARCH ISSUES

Although the use of Information Engineering concepts can be beneficial for the development of SIS, some limitations restrict these benefits.

Boutin (5) has identified several SIS problems not addressed by current Information Engineering system design methodologies, techniques and tools for non-spatial information systems. These problems were presented in the form of 50 questions concerning the following topics:

1) How and when to make the inventory and analysis of cartographic data, including lineage?

2) How to make a spatial database schema considering geometric entities, attributes and relationships, geometric descriptions of spatial entities, implicit spatial relationships, spatial meta-data, cartographic background, semantic incoherence introduced by geometric descriptions of spatial entities, cartographic generalization, etc.?

3) How to model spatial processes?

4) How and when to select base maps, reference systems, spatial precisions and spatial referencing techniques and what is their influence on implicit spatial relationships?

5) How to select GIS software?

6) How to build cartographic data dictionaries.

We can add in this list the absence of appropriate tools for fast GIS prototyping (e.g. cartographic 4GL map painter, spatial application code generator).

If we really want to take advantages of Information Engineering concepts and technology, new methodologies, techniques and tools must be built or existing ones must be extended to better include the cartographic component of a SIS (6).

Among the previously mentioned limitations of non-spatial Information Engineering methodologies, techniques and tools, some have already been addressed in the SIS community. However, certain aspects remain research issues.

For example, the research group of the Spatial Information Systems Laboratory of the Geomatics Center at Laval University is currently working on several topics. Beside Boutin's work which was completed in 1988, research is being done on building an extension to the Entity/Relationship Formalism to better model spatial data. Preliminary results of this work were presented by Bédard and Paquette (2). Following the completion of this project, the building of a CASE tool for spatial data modeling is planned; this tool would cover all steps from conceptual data modeling to automatic code generation for spatial database creation. The results of this long term research should solve most of Boutin's second item.

A second research project relates to the building of a tool for the inventory and analysis of the cartographic data of an organization. Such a tool would formalize this step of the development of a SIS and provide techniques to describe the lineage of maps. The result should be amenable to a computer form for an eventual CASE tool. This research would solve the first item of Boutin's list.

A third research project involves the analysis of map compatibility based on several criteria (spatial, temporal, descriptive, format) and should result in a tool providing criteria and levels of compatibility. This project would complete the previous project.

Finally, some attempts are being made towards building cartographic data dictionaries.

Thus, these examples show that to make better use of Information Engineering for SIS, good spatial inventory, diagramming and analysis techniques and tools still have to be built. The results must be rigorous, methodical and provably correct as they must be amenable to a computer form.

It is only when such techniques and tools have been created, according to the Information Engineering approach, that we will be able to develop CASE tools specialized for SIS. This will allow for the automation of an increasingly number of steps in the development of SIS; thus accelerating, and improving, such development.

While developing these new techniques, we must take into consideration the capabilities offered by the computer. For example, it is easier to use rigorous techniques if the computer performs several checks in place of the designer. Thus, much more powerful tools and techniques can be used with the computer than is possible otherwise. In fact, while Information Engineering began by computerizing existing tools and techniques, it has gone onto improving these tools and techniques and, indeed, to building new and better ones which take full advantage of the possibilities of computers. As stated by Martin and McClure (11), "the most important trend is the move in techniques that are the basis for CASE... The challenge today is to harness the computer as powerfully as possible in the entire process of developing systems".

CONCLUSION

As new and powerful GIS software makes it more tempting to build spatial information systems, it becomes increasingly important to use the right tools, techniques and methodologies as suggested by the discipline of Information Engineering.

The arrival of CASE tools which automate several steps of the development process, as well as the mixing of prototyping techniques with system design methodologies meet, *today*, some of the requirements for developing Spatial Information Systems (which are often large, complex and uncertain).

However, important developments are still required, especially in the adaptation for specific applications such as Spatial Information Systems. Research and Development is needed such that in the not too distant future, SIS developers will use Spatial CASE tools to automate the development of SIS.

When we consider today's need for all kinds of earth-related information (e.g. environment, land use, natural resources), it becomes important to apply Information Engineering research to SIS. We must move towards Spatial Information Engineering if we want to accelerate the development of SIS and improve the results.

ACKNOWLEDGEMENTS

This research has been made possible by a grant from the Natural Sciences anti Engineering Research Council of Canada (NSERC).

REFERENCES

(1) Alavi M. "An Assessment of the Prototyping Approach to Information Systems Development", Communications to the ACM Vol.27, No.6, June, 1984, pp.556-563.

(2) Bédard Y. and F. Paquette. "Extending Entity-Relationship Formalism for Spatial Information Systems", Proceedings of AUTO-CARTO 9, American Congress on Surveying and Mapping, American Society of Photogrammetry and Remote Sensing, Baltimore, April 3-7, 1989, 10 p.

(3) Bédard Y., Prince J. and G. Boutin. "Information Engineering for the Development of Spatial Information Systems", Challenge for the 1990s: GIS Geographic Information Systems Canadian Institute of Surveying and Mapping, February 27-March 3, 1989, Ottawa, 10 p.

(4) Bouhot J.-P. "Analyse des méthodes. La lettre de Jean-Pierre Bouhot", L'informatique professionnelle, Bouhot et Le Gendre publications, No. 52, March, 1987, pp. 3-12.

(5) Boutin G. "Étude de l'applicabilité d'une méthode traditionnelle de conception de système d'information dans le contexte d'un système d'information à référence spatiale" M.Sc. Thesis, Department of Geodetic Sciences and Remote Sensing, Laval University, April, 1988, 206 p.

(6) Boutin G. and Y. Bédard. "On the Use of a Standard "Information System Design Methodology" for the Development of a Land Related Information System", Third International Seminar on Trends and Concerns of Spatial Sciences, Department of Geodetic Sciences and Remote Sensing, Laval University, June 6-8, 1988, 23 p.

(7) Burns R.N. and A.R. Dennis. "Selecting the Appropriate Application Development Methodology", Database Fall, 1985, pp. 19-23.

(8) Campbell G. J. "Integrated Spatial Information System Design Through Information Engineering", Third International Seminar on Trends and Concerns of Spatial Sciences, Department of Geodetic Sciences and Remote Sensing, Laval University, June 6-8, 1988.

(9) Dubreuil S.H. "Utilisation des méthodes: on y vient...", Le monde informatique numéro spécial 288, juillet-août, 1987, p. 42.

(10) Humphries, B. "The Crisis in Land Management", The Decision Maker and Land Information Systems, Papers and Proceedings from the Fédération Internationale des Géomètres International Symposium, Commission 3, A.C. Hamilton and J.D. McLaughlin Ed., Edmonton, October 15-19, 1984, pp. 8-18.

(11) Martin J. and C. McClure. "Structured Techniques: the Basis for CASE, Revised Edition" Prentice Hall, 1988, 776 p.

(12) McClure C. "The CASE for Structured Development", PC Tech Journal for Systems Developers and Integrators Vol. 6, No. 8, August, 1988, pp. 51-67.

(13) Watterson K. "Making a Case for CASE", Data Based Advisor Vol. 6, No. 8, August, 1988, pp. 132-136.