THE STRATEGY OF IMPROVING INFORMATIONAL SYSTEMS THROUGH ECONOMIC APPLICATIONS

Emilia VASILE, PhD Professor
Athenaeum University, Bucharest, Romania
rector@univath.ro

Dănuţ-Octavian SIMION, PhD Associate Professor
Athenaeum University, Bucharest, Romania
danut_so@yahoo.com

Abstract: *The paper presents the strategy of improving informational systems in economic applications. The architecture of the information system is the generic solution for the data processing processes of data integration and processing. The company's global IT system breaks down into subsystems, each of which covers a distinct field of activity. In turn, each subsystem breaks down into applications, each of which covers a distinct activity within the domain. In defining the architecture of the information system there are three strategies that have crystallized over time. The downward strategy: top-down goes from the principle of complex computer system decomposition to components with lower complexity, the upward strategy: bottom up promotes the initiative at every level of management (accounting, commercial, production, etc.) without a framework solution and a defined architecture for the global IT system at the organization level and the mixed strategy that is a combination of downward strategy with bottom-up strategy taking their strengths into account. The best strategy includes a mixed of the standard strategies and an important role plays the economic applications that include algorithms that improve the business logic.*

Keywords: *business strategies, business environment, programing algorithms, information systems, business alternatives, business logic*
JEL Classification: *C23, C26, C38, C55, C81, C87*

1. Introduction

Management information systems are defined by two approaches:

a) starting from the information and its support;

b) starting from the function that the management information system has to accomplish.

In the first case, management information systems represent all the information used in the company, the means and procedures for identifying, collecting, storing and processing the information.

The second approach to defining management information systems starts from its purpose, namely to provide the information requested by the user in the desired and timely manner in order to substantiate the decisions.

Management Information Systems (MISs) imply the definition of: management domains, data, models, management rules.

The management domains correspond to each of the homogeneous activities carried out within the company - research, development, commercial, production, personnel, financial and accounting - taking into account the interactions between them. Moreover, the approach of these domains is done in a hierarchical vision leading to the identification of the following levels:

• Transaction in which elementary operations are performed;

• Operational where current operations are taking place, the decisions made at this level are current, of routine;

• Tactics corresponding to control activities and short-term decisions;

• Strategic feature of long-term decisions and / or global engagement.

Data is the „raw material" of any management system. All data reported and processed regardless of their nature, their formal or informal character, or the media on which they are located are taken into account (Schneidewind 2015; Lloyd 2016).

2. Management models and informatic systems

Management models summarize the domain's own procedures. We can exemplify through the model:

• Accountant, specific to the financial-accounting field;

• Manufacturing technology specific to the field of production;

• Sales specific to the commercial domain.

Management rules allow data processing and use of information in accordance with the objectives of the system.

Within a company with production and / or commercial activity, the following management rules can be identified:

- the supply is made when the stock actually falls below the standard stock;
- the material evaluation is carried out according to the FIFO method;
- a raw material is stored in one or more management;
- for second quality products the price is reduced by 5%, etc.

In the case of a bank, the following management rules can be specified for the current account information system:

- minimum balance;
- Payments are made within the balance limit;
- interest calculated for sight accounts is 11% per year;
- Up to two people with the right of signature can be registered.

Through the notion of domain, we come to the concept of a functional management subsystem based on functional criteria, on which the other two concepts are grafted: the management model and the management rules.

The management information system assures and provides the information requested by the user, using the IT tools, to substantiate the decisions regarding a certain area within the firm (Bakos 2016; Porter and Millar 1985).

Current IT systems are integrated systems. They are characterized by the application of the principle of the unique data entry and multiple processing in accordance with the specific information needs of each user.

Integrated accounting is characterized by a unique introduction of data taken from primary documents that update a single accountancy database that will then be exploited to ensure both the specific financial accounting and management accounting tasks, - thus, the processing requirements of all users.

There are some approaches in the development of information systems:

In the development of a computer system one can choose one of the following solutions:

- a centralized computer system
- a decentralized computer system

The centralized computer system is characterized by the fact that the whole process of data storage and processing, as well as the development of the system, takes place at a single location where there is a single computing system, usually a mainframe, which stores a base unique data as well as all application programs. Users interact with the system via terminals (which act as thin clients).

The advantages of centralization are represented by:

- effective control over the use and development of software;
- control over data security and integrity;
- sharing hard, soft and data resources among users;
- eliminating the risk of hard and soft incompatibility within the system;
- Easily promote standards (technical, design, procedural, etc.) at the level of the whole system;
- providing the services requested by the users through the power of calculating the central system (mainframe).

Disadvantages of centralization are the following:

- the „fall" of the computing system blocks all users;
- Alteration of data and programs, whether void or accidental, affects all users;
- the system may prove slow and inflexible to users' needs, often insufficiently adapted to local or group needs of users;
- can achieve a long response time in case of simultaneous requests of multiple users. The decentralized information system is characterized by the fact that the data, software and power of

the calculations are dispersed in different locations (even geographically dispersed) of the organization. Processing takes place on independent personal computers or on local networks.

Advantages of decentralization:

- data is stored and processed locally;
- software is better suited to local needs;
- Hard, soft or database failures at a location do not affect other locations.
- the system configuration can be tailored to the needs of different departments within the organization or even local users;
- greater autonomy and motivation at the local user level.

Disadvantages of decentralization:

- high risks related to hard and soft incompatibilities between different locations;
- the inherent appearance of duplications of data and software in different locations;
- the difficulty of realizing complex projects at the local level;

- the risk of fragmentation of IT policy;
- higher costs than the centralized system.

The current trend is net-oriented towards decentralization, which must be done in such a way that:

• All responsibility and authority for the decentralized functions of SI to belong to local management;

• Ensure alignment with the standards used at the organization's overall SI level;

• at central level to be achieved:
- elaboration of strategy at the whole SI of the organization;
- communication management within the organization's local network;
- data management;
- disaster recovery.

Today, the architecture promoted in decentralized systems is the client-server architecture characterized by the fact that the applications and the data available to the users are dispersed on the different hardware depending on the number of users to access and the required computing power (Schneidewind 1987; O'Brien and Marakas 2015).

- Departmental applications that:
• running on the departmental server;
• exploits at department level data stored on its server;
• are shared by users of the same department;
- Organizational apps that:
• running on the central server;
• exploits data of general interest stored on the central server;
• are shared by users of multiple departments;
• requires high processing power.

3. Principles of design and implementation of management information systems

Conducting a rigorous and efficient design and implementation of management information systems requires the following principles to be observed:

1. The global approach to the problem solved;
2. Using a unitary methodology in the design and implementation of the information system;

3. Application of the most modern solutions and methods of designing and implementing the information system;

4. Structure of the IT system taking into account the organizational structure within the company.

5. Direct participation of the future beneficiary in the analysis, design and implementation of the information system.

Such participation ensures that the design specifications and the gradual validation of the solutions proposed by the designer are clearly formulated, all of which ensure a product that fully complies with the user's requirements;

6. Compliance with the legal framework. In the case of management information systems, it is mandatory to record, compute the indicators and prepare the synthesis work in accordance with the regulations in force.

7. Developing computer systems for the resources available to the user;

8. Since the software is subject to change, this change must be anticipated and controlled;

9. Compromises are inherent in software development and must be explicit and documented.

Specialty studies have attempted to highlight the success factors in running the software projects. The Standish report, for example, places as prime success factors:

- End user involvement
- Support of executive management
- Clarity of requirements
- Planning

Conceptual modeling of data

Entity-Association Model (EA)

To define the conceptual model of data, we use intermediate models that are used as a support for a design methodology. A conceptual model is a set of concepts and rules to combine these concepts, allowing the representation of the reality circumscribed to the field subject to computerization.

The models used are called semantic models and aim to provide the real world with the concepts offered. Semantic models use abstractions representing the real world as a collection of entities and relationships established between them. Most models allow the definition of restrictions describing the static, dynamic or even temporal aspects of entities.

The entity-association model (EA) that we will continue to use for defining the conceptual model of data is also a semantic model.

The EA model seeks to obtain a fair representation, using specific concepts, of the reality (the problem to be solved to be computerized). This representation of the real world will be achieved by abstaining from any restriction either computer or organizational. Starting from the semantics of real world objects and the relationships established between them, the EA model also serves as a means of communication between the modeler (computer) and the future user of the system (the beneficiary of the information system) describing the reality subjected to modeling according to his / her own perception.

The basic concepts of the Entity Association model:

Entity is an object of the modeled reality characterized by its own existence, with its own identity (which makes it identifiable with respect to the other objects of the same type) and a number of characteristics that express its properties. For example, we propose to you as a field of study the management of policies concluded by an insurance company.

In modeling, interest focuses on defining the types of entities belonging to the problem to be solved, and not on the entities representing the achievements of the types of entities. Instead of the entity-type notion, some authors use the class of entity concept. The abstract model that the EA model gives us is based precisely on these generic types of entities and the relationships established between them (Spence 1975; Porter and Millar 1985).

Type of entity is a generic concept designating the set of all entities presenting the same constructive characteristics. Examples: product, order, employee, student, contract, insurance policy, bank deposit, exchange order, etc. This time, the type of product entity designates all of the products in the company's catalog, described products based on the same common features: product code, name, unit of measurement, date of approval, percentage of VAT.

Attribute defines a distinct property of an entity. Each attribute has a range, that is, a set of admissible values. In an entity there are achievements corresponding to the defining characteristics of the entity type.

Attributes can be broken down by several criteria:

a) After complexity, the attributes are:

• Elementary (simple) whose achievements can not be broken down (example: monetary unit, unit price, student's student number, employee's mark, etc.).

• decomposable (complex) whose achievements are decomposable (eg: calendar date - can be broken down by day, month, year, address - can be broken down in the street, number, classification code of a fixed asset, etc).

b) After the achievements that the attributes can represent can be:

• mandatory (must necessarily represent an achievement that corresponds to the NOT NULL syntax - any realization) and • optional (are attributes that may not show any value within an entity (eg the attributes: telephone, fax, e-mail - not everyone has a phone, fax, e-mail address).

• monovalents: Attributes that represent a single value within an entity (example: student name, matrix number, date of birth, personal numeric code, etc.) and • multivaluers: attributes that represent multiple achievements within the same entity

The modeling problem is often very complex within it, identifying simple, composite and / or complex objects. In the EA model, the real world objects correspond to entities, and the entities defined by the same characteristics form a type of entity. This means that simple objects will correspond to one type of entity in the conceptual model of data.

Restrictions on integrity define the requirements that data must observe to be fair and consistent with the reality it reflects.

Integrity restrictions are a way of integrating data semantics indirectly into the association entity model that they enrich.

Restrictions on integrity concern:

• the values that attributes of entities and associations can take;

• entity identifier values;

• the roles played by the entities in the associations in which they participate;

• associations established between entities.

Integrity restrictions may be static (permanently verified) or dynamic (regarding the evolution over time of data). For example, restrictions on the number of the management, the type of the deposit and the units of measurement are static. The restriction on the VAT rate is dynamic and may change over time in line with the tax provisions in force.

Domain restrictions

The domain, as a set of values that an attribute can take, can be defined by a property (a condition for an attribute or a group of attributes) by specifying a range of values or enumerating the set of admissible values.

Domain restrictions are conditions (rules) that pertain to the set of admissible values for an attribute within its type or domain. Restrictions may refer to the achievements of an attribute belonging to the same entity or association, in which case intrastate restrictions or attribute belonging to different entities and / or associations are called, in which case they are called interrelationships.

Structural Restrictions Identifying Entities

Each entity will need to be unequivocally identified. This requires the entity identifier to take unique values different from NULL (NULL means that no value has been assigned, so NULL is different from zero or space).

In defining the EA model, we may encounter more specific cases of identifying entities: We can not define an identifier in the form of an attribute / group of attributes for a particular type of entity. Example: For each employee of the company, the monthly presence is recorded by keeping the hours actually worked, the absences, the sick leave hours. Identifying Entities Presence is done by the role Performed by the Employee Entity in the Associa tion Register The entity identification of entities can only be achieved if the association in question is not cyclical (unary) and the cardinality of the identified entity-association couple is 1.1 and the cardinal value of the couple identifier - association is 1.1 or 0.1.

The identification of an entity can be accomplished by one or more of its own attributes along with the role played by another entity within the association.

In addition to specifying the number, the policy date, the insured amount and the specification of payment premiums with the date of the payment and the payment amount, we consider a life insurance policy (we consider that the amounts are different from one maturity to the next). Each policy is identified by number, uniquely assigned. For the type of Maturity entity, the Maturity date attribute can not be an identifier because multiple policies have the same payment deadline on the same date. Identification of the Prime Insurance entities will be achieved through the values of their own attribute Date of Maturity and the Role Playing role of the entity Insurance in association (Bakos 2016; Spence 1975).

Role integrity restrictions

In defining the association, I emphasized that it expresses the established link between different entities, each of which plays a role. Starting from the roles played by entities within the associations, we can define a series of integrity restrictions, namely: equality, inclusion and role exclusion.

Restriction of role inclusion. The role-inclusion restriction states that if an E1 entity that plays the r1 role in the A1 association will also have to play the r2 role in the A2 association. It follows that the role r1 includes (by inclusion) the role r2.

An example is the CDM sequence developed for an insurance company. Clients conclude insurance policies and receive indemnity to produce the insured risk if they are up to date to pay insurance premiums. There is an inclusive inclusion restriction between the compensated role and the paid role. It is also a restriction of integrity on the compensated and terminated roles the policy entity plays.

Role Restriction

Equality of roles implies that the role-sharing restriction between roles is reciprocal. The persons who have a role in the financial administration, so they play the role of the owner in joining Are, means that they have fiscal obligations, so they play a role as a taxpayer in the association. It is true, and reciprocal is valid: everyone who has tax obligations must have open a role. But not all people are taxpayers, so there are entities of the type Person who does not play the role of the Owner in association Has (Spence 1975; Lloyd 1987).

Ex: In economic applications it could be used Feature Templates (template functions)

You can define functions that work on different types of data, the function code being written once using a generic date type. In this case, a pattern is created that automatically generates the functions of each top.

The syntax for declaring a template function is:

template <[list of types] [, [list_arguments]]> statement

where:

- list_types - is a list of date types, indicated under
form class id;
- list_arguments - is a list of arguments given in the form
id_type_name.

The function that calculates the minimum of the elements of a vector will be defined, the vector elements being of a character type (whole on a snake), long or short, or floating point:

```
#include<stdio.h>
#include<conio.h>

// definire template
template <class T>
T minim (T a[], int n)
{
      T m=*a;
      for(int i=1;i<n;i++)
      {
            if (a[i]<m) m=a[i];
      }
      return m;
}

// the main program calls the minimum function for two types of
data: int and double

void main()
{
      int v1[]={5,1,8,-9,67};
      double v2[]={3.79,4.18,9.6};
      printf("\nMinim is v1 este %d",minim(v1,5));
      printf("\nMinim is v2 este %lf",minim(v2,3));

 printf("\n");
 getch();
}
```

For example is the program that, depending on the number of values read from the keyboard, selects and automatically launches one of the functions:

a. f1 = -1, if no value is read;
b. f2 = x2, if a value is read;
c. f3 = x * y if two values are read;
d. f4 = x + y + z if three values are read;
e. f5 = x * y + z * t if four values are read.

Selecting the functions is done by checking the value returned by the scanf function, ie the number of parameters read correctly from the keyboard.

```
#include<stdio.h>
#include<conio.h>
int f1(int a, int b, int c, int d)
{ return -1;}
int f2(int a, int b, int c, int d)
{ return a*a;}
int f3(int a, int b, int c, int d)
{ return a*b;}
int f4(int a, int b, int c, int d)
{ return a+b+c;}
int f5(int a, int b, int c, int d)
{ return a*b+c*d;}

//programul principal
void main()
{
 int (*pf)(int, int, int, int);
 int v, x,y,z,t;

switch(scanf(«%d %d %d %d»,&x, &y, &z, &t))
{
 case 0: pf=f1;break;
 case 1: pf=f2;break;
 case 2: pf=f3;break;
 case 3: pf=f4;break;
 case 4: pf=f5;break;
 default: break;
}
 v=(*pf)(x,y,z,t);
 printf(«\n Rezultat=%d»,v);

 printf("\n");
 getch();
}
```

These types of algorithm helps to resolve different kinds of economic problems and through the logic implemented it might treat many scenarios. In many cases a special function resolved a specific kind of problem that is included in a more complex systems that is designed for economic applications (Schneidewind 1987; Porter and Millar 1985).

4. Conclusions

Through the implementation of mathematical models and the use of computing in specific activities, the computer system delivers enhanced valences to the information system in quantitative and qualitative terms. It is about increasing the computing capacity in terms of the volume of data processed and the operations performed, increasing the accuracy of the information, increasing the efficiency, complexity and completeness of the reporting-information situations, etc. All this leads to a greater approximation of the decision maker to the phenomena and economic processes he has in mind, with the many positive economic aspects deriving from it (Bakos 2016; Spence 1975). Regarding the relation between the information system and the informational system, it can be appreciated that the information system tends to equalize the dimensions of the information system, but it does not have the same scope as the latter, because within the information system there can be activities that can not be fully automated (Porter and Millar 1985; O'Brien and Marakas 2015). Any economic application requires a number of adaptive algorithms to resolve issues that are not part of any pattern. The flexibility of the algorithms provides extra flexibility in the complex approach of situations where all the variables or all the data of the problem are unknown.

References

Bakos, J. Y. (2016). Recent applications of economic theory in Information Technology research. *Decision Support Systems Journal*, 8(5), pp. 365-386.

Lloyd J.W. (1987). *Foundation of Logic Programming*, 2nd ed. Berlin: Springer-Verlag.

O'Brien, J.A. and Marakas, G.M. (2015). *Management information systems*. NewYork, NY: Mc-Graw Hills.

Porter, M.E., and Millar, V.E. (1985). How Information Gives You Competitive Advantage. *Harvard Business Review*, 63(4), pp. 149-160.

Schneidewind, N.F. (1987). The State of Software Maintenance. *IEEE Transactions on Software Engineering*, 13(3), pp 303-310.

Spence, A.M. (1975). The economics of internal organization: An introduction. *Bell Journal of Economics*, 6(1), pp.163-172.