REGISTERS AND REGISTER-BASED INFORMATION SYSTEMS

Albertas Čaplinskas
Institute of Mathematics and Informatics
Akademijos 4, 2600 Vilnius,
Lithuania
E-mail: alcapl@ktl.mii.lt

Olegas Vasilecas
Vilnius Gediminas Technical University
Saulėtekio al. 11, 2054 Vilnius,
Lithuania
E-mail: olegas@rasa.vtu.lt

Abstract.. The main purpose of this paper is to consider the role of registers in modern information system engineering. It discusses the notion of register, proposes the architecture of the register system, the classification of registered units, and a register-based approach to IS engineering.. The paper also introduces some concepts and terms as the basis for further discussion how to build, maintain, and use register systems.

Key words: public registers, institutional registers, information systems, information system engineering.

1. Introduction

The main purpose of this paper is to consider the role of registers in information system (IS) engineering. Registers serve as a foundation for many information systems in public administration as well as in the enterprises, companies, and universities. A modern register is a complex system composed of several data bases, software programs, services, organisational procedures, and other components. Registers used by an information system or a group of such systems should be integrated and considered as a coherent whole. In addition, any register, wholly or partly, may be a projection of a higher-level register (up to the national level registers). So we deal with a many-level system that is too complex to be designed, implemented, and maintained as a whole one. On the other hand, standards, architectures, methods and techniques of building registers and register-based IS are still a challenge [1], [2], [3], [4]. The lack of a sound conceptual basis and unified terminology is faced in many practical problems.

This paper introduces some concepts and terms as the basis for further discussion, first of all in the Baltic states, because integration efforts in this region will be hardly fruitful without common accepted standards how to build, maintain, and use register systems. It is organised as follows. Section 2 discusses the notion of register. Section 3 proposes the architecture of the register system. Section 4 discusses the ground cluster hypothesis. Section 5 proposes the classification of registered units according

to their addressing data. Section 6 discusses a register-based approach to IS engineering. Some concluding remarks are presented in the last section.

2. Register system

Let us consider the notion of register. A register is a system provided by an authority and intended to register, store, maintain, and present the data on so called *registered units*. It must be established by an institutional act (law, decree, order, etc.) that defines which units and properties are to be registered and provides the registration rules, procedures, and services to be supplied by the register. There are two kinds of registers - *primary registers* and *secondary registers*. A primary register is the one that registers an unregistered unit. A secondary register is a register that registers a role of unit already registered in the primary one. A registered unit may be an object, an object state, a role, a relation, a process, an activity, an event or other entity whose being, usage, occurrence or other aspects are managed, controlled or regulated by an institution (state, agency, enterprise, university, etc.). If several kinds of associated units are registered, then one kind of unit must be declared to be basic. We call the basic kind of a registered unit a *registration object*. Usually the registration object is an object and supplementarily registered units are events and/or relations. An example is the land parcel register that registers parcels, ownership relations, sell-buy events, etc. A parcel is the registration object of this register.

We propose to divide the data of a registered unit into *registration data*, *basic data*, *descriptive data*, and *application-oriented data*. *Registration data* include the registration date and place, registrar's name, and other official data about registration. *Basic data* are those identifying a registered unit, describing its localisation in space and time, and defining its relationship with other registered units. The basic data must be registered necessarily. *Descriptive data* are those that serve for many business activities as a basic information source about objects, processes, events, and other business entities. That is why descriptive data must describe global, essential, activity independent registered unit properties only. They may be registered as obligatory or collected by the register in another way. *Application-oriented data* are those that are of local interest, used for some activities or in some departments only. They are not the subject of registration though the requirement to register application-oriented data necessarily is a widespread mistake in IS engineering.

3. Register's architecture

Let us consider the architecture of the register system. We propose the architecture that includes registrar's DB, original DB, central DB, local DB, and application-oriented DB (Fig. 1).

A *registrar's DB* is a database used to collect and verify primary data. The register has as many registrar's DB as registrar's workplaces are established in this register. The data collected in registrar's DB are transferred to original DB according to the regularity provided by the institutional act.

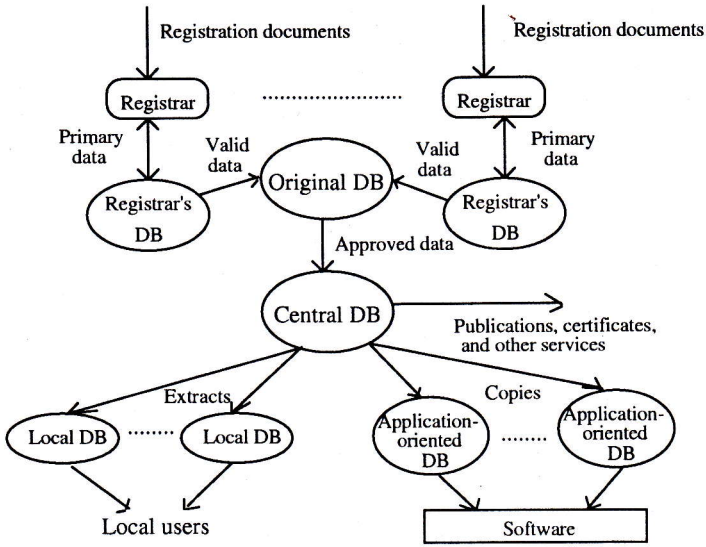

Fig. 1. Register system

An *original DB* is a historical-descriptive database used to store authentic (in legal sense) registered data. It includes the registration data, basic data and those descriptive data that are registered obligatory. This base saves all registration records in chronological order. The register has only one original DB. It may be distributed. Register's certificates, copies, and extracts are legally valid only in case they conform to the data in the original DB.

A *central DB* is a descriptive database used as the main copy of the original DB. In the central DB only actual data are stored. This base is used to publish the register and to produce certificates, copies, and extracts. It includes basic and descriptive data. Thus, the central DB stores the application-independent data of global interest only. It means that for institutional acts to be correct it should also provide the registration of only application-independent data of global interest.

The register may establish any number of local DB or application-oriented DB. A *local DB* is a database used in a region, branch, department, or an other local unit of the institution. It is created on the basis of a copy or an extract from the central DB.

As usual, the local DB also includes application-oriented data that are of local interest only. *An application-oriented DB* is a data base that is established to satisfy information needs of a group of application systems. On the national level, an application system is an IS. On other levels, it is a subsystem of the institutional IS. Usually an application-oriented DB is created on the basis of a copy of the central DB. In addition it includes some application-oriented data. The register's duty is to actualise the registered data in all its local and application-oriented DB opportunely.

It should be noted that attempts to join original, central and registrar's DB, or central and applications-oriented DB are widespread mistakes made in the design of register systems. The decision to combine several DB increases the coupling of IS components and, as a result, causes other undesirable consequences. On the other hand, sometimes it is purposeful to simplify the proposed architecture for the small and simple registers.

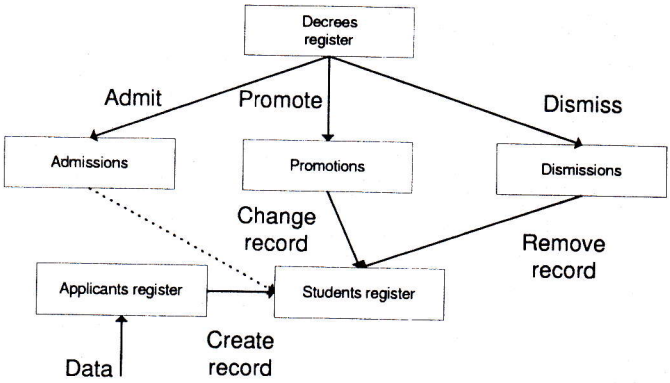

Fig. 2. Register interaction example

Events that change the states of the registration object play a particular role in the register architecture. Events may be registered in a different way:

1. Events cause changes in registered data but they themselves are not the subject of interest and are treated as supplementary units. In this case, events and changes form the registration history stored in the original DB, while in the central DB only actual versions of the registered data and the reference to the last event are stored. An example is the study module register.

2. Events themselves are the subject of interest. In this case, the register is composed of a number of subregisters: one to register instances of the registration object (main subregister), and one or more to register events. Some subregisters may be remote. Each subregister is established by an autonomous institutional act and has all the

databases presented in Fig. 1. An example is the natural person register. Persons, births, marriages, and deaths are regarded as autonomous registered units and registered by autonomous subregisters.

3. Events and units the state of which they change are registered by autonomous registers. The event register registers all events that are caused by an agent or a structure of subordinate agents. Events are classified according to the registers the state of which they change. The categories of events should actually be regarded as subregisters of those registers. An example is the university decree register (Fig. 2). The decree may be published by the rector, dean or some other university authority. Decrees describe events that change the state of students, staff, and other registers. It should be noted that events are often described by documents. Sometimes events and documents are even indistinguishable.

4. Events are not registered on the whole and their occurrence is checked by data derivation procedures. For example, the natural person becomes of age when he is eighteen years old. This fact is usually tested by an attached software procedure.

5. Sometimes the subject of interest is events only. The changes made by events and even the changed entity are out of the scope. In this case, we have a proper event register. Typical examples are the accident register or the solar eclipse register.

What kind of event registration is used depends on the event legal status. In different cases different architectural solutions should be used. As a rule, today the event registration does not imply automatic changes in the main subregister. Changes are registered and approved independently after the appropriate events are registered. It seems, however, that the network technology today allows us to use the object-oriented technology and design the register system so that changes be propagated automatically (Fig. 2). Software that changes registered data can be designed and implemented as a set of methods, and these methods should be regarded as an inherent part of the register system. At present, such a software is usually designed and implemented as a part of some IS subsystem. Such approach is not the best solution because it increases the coupling degree between registers and the application software.

4. Ground cluster hypothesis

As a rule, an institutional IS is based on tens of registers that include hundreds of cross-references. For example, the Vilnius Gediminas Technical University IS [9] is based on more than 30 registers. To integrate these registers, it is necessary to develop a lot of classification, codification, and identification standards. It is a very complex task. However, registers mirror the real world where business activities usually focus only on few registered units. Our hypothesis is that, as a result, it is

often possible to identify several relatively closed groups of registers called *ground clusters* (Fig. 3).

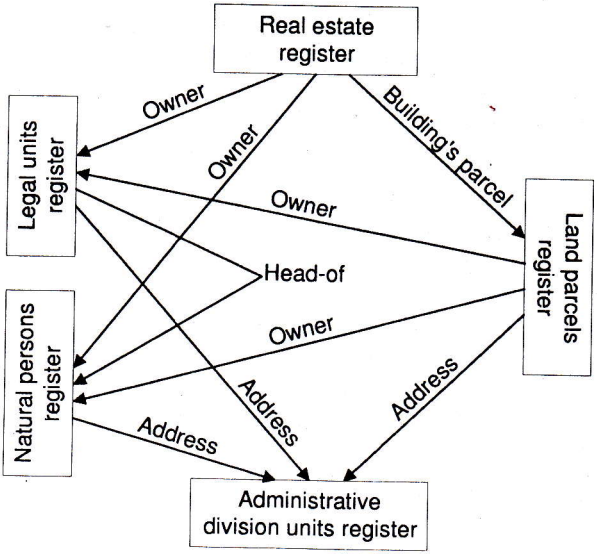

Fig. 3. Ground cluster example

The ground cluster is closed in the sense that cross-references are made only between registers belonging to the cluster. A register may belong to several ground clusters. Other registers have not cross-references. They refer to ground clusters only. Thus, it is possible to integrate registers inside the ground clusters and start their implementation. Other registers can be integrated into the system step-by-step.

In the Vilnius Gediminas Technical University IS four ground clusters were identified. The first cluster includes the staff, student, department, module, and study program registers and the classifier of sciences. The second cluster includes the staff, post (appointed or to be appointed), and department registers and the classifiers of positions, and of scientific degrees. The third cluster includes the staff, and research, and the classifier of sciences. The fourth cluster includes the staff, department, real property, appliances, and room registers and the classifier of property and appliances.

5. Classification of the registered units

To locate registered units on a digital map, it is important to develop the classification of registered units according to their addressing data. The classification of registered units according to their reference attributes facilitates the register integration problem.

Let us consider the classification of registered units according to their addressing data more in detail. First of all we divide registered units into *localizable* and *unlocalizable* units. Localizable units are those that are addressed in the space. Unlocalizable units include the following subcategories:

1. *Properly unlocalizable units*: abstract units that are nowhere and, on principle, cannot be addressed in the space.
2. *Units, supposed to be unlocalizable:* printable units (texts) that are stored in the register as its descriptive data.
3. *Indirectly localizable units*: tangible units that are possessed by somebody and stored in the place that is known only to the unit possessor.

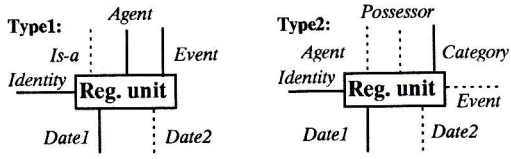

Fig. 4. Unlocalizable units (dot lines mark facultative references).

It should be noted that printable units are sometimes also regarded as units that are addressed in the space. In this case, the text stored in the register has the status of a copy and the original text is stored somewhere, for example, in a folder. This folder is the unit location.

The category of unlocalizable units splitss up into two types (Fig. 4). To type 1 belong such properly unlocalizable units that refer only to events and are addressed in time by the interval *Date1*, *Date2*. They may also form an *Is-a* hierarchy. The *agent* is a subject that inspires occurrence of an *event* that acts as a constructor and puts the unit into existence. A typical example of type 1 units is categories (e.g., position categories). To type 2 belong such unlocalizable units that can be classified using one or several classifiers. They may be simple units or aggregate. If a registered unit is an aggregate, the register must have a *part-of* relationship subregister. The *possessor* is a subject that possesses the unit (i.e. owns, has a power of ordering, disposing, or managing, or is officially appointed to keep it in good repair or working order). Typical examples of type 2 units are the study programs and modules, publications, intellectual property (copyrights), and the certificates given out by the institution (e.g. bachelor's diploma).

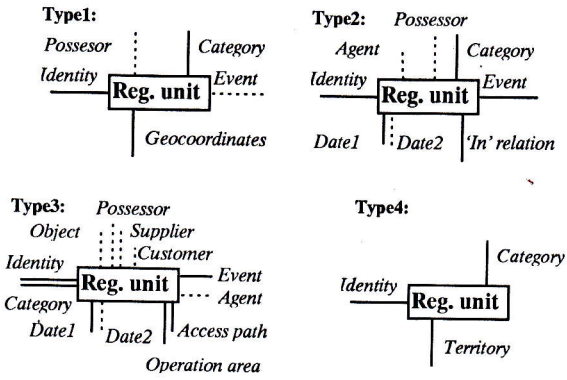

Fig. 5. Localizable units (dot lines mark facultative references).

The references to locations arean especially important sort of references. A location is a place where the registered unit was, is, or is to be. It may be a point or a region and even a line. The registered unit can refer to the location in four different ways and, consequently, the localizable unit category breaks up into four types (Fig 5).

1. *Directly addressed units:* units that refer to the location by geocoordinates A registered unit may refer in such way to points as well as to regions and lines. Typical examples are land parcels (regions), rivers (lines), and monuments (points). Some accidents, crashes, and other events can also be addressed as type 1 units. A possessor example is an institution that is responsible to keep the monument in good repair.

2. *Indirectly addressed units*: units that refer to the location using the reference to another unit and the *in* relation. A typical example of type 2 units is buildings. A building refers to a parcel as to its location (Fig. 3). Some accidents, crashes, and other events can also be addressed as type 2 units.

3. *Units, addressed by a post or some other address*: units that refer to the location using an access path (e.g., *country.region.city.zip_code.street.building.room*, *disc.directory.subdirector. ... .subdirectory.file*, *department.subdepartment.room*, *room.bookcase.shelf.folder*, etc.). Sometimes the access path includes the reference to the deliverer (e.g., a zip code). Only the points can be addressed in such way. Using a shortened access part, regions can be addressed, too (e.g., country.region.city). In the extreme case, the search path degenerates to a path of length 1 and addresses a point only (e.g., folder number). Typical examples of type 3 units are natural and corporate persons, correspondence and decrees. Object examples are units 'about whom' there are decrees or correspondences. Possessor's example is a person to whom an official letter is addressed. The operation area is a region usually

described by a shortened access path. An example is the territory served by a post office. Suppliers and customers are used, for example, to describe contracts. Agent's example is a provider of services (e.g. post office).

4. *Distributed registered units*: units that refer to the location using a raster. Typical examples of type 4 units are nationalities, plants, minerals, etc. As a rule, these units can also be addressed as type 3 units.

Each type of registered units may be classified further according to the typical combinations of reference attributes. We identified fifteen typical patterns. However, it is impossible to describe them in so brief a paper.

6. Register-based approach to IS engineering

As a rule, the system analysis starts from the gathering of business knowledge. The register-based approach assumes that registration is an important method of collecting facts about business entities. The proposed approach considers an integrated register system (IRS) as an autonomous part of the institutional IS that can be designed and implemented first of all.. The problem-oriented and task-oriented IS subsystems must be designed and implemented at the same time with some delay or after the whole IRS is implemented. Since the IRS usually covers a significant part of institution information needs, it can be regarded as the first version of the whole IS.

We suggest dividing the IRS development process into six major tasks:

Task 1: *Identify entities have to be registered.* The objective of this task is to identify which entities are or must be registered in the institution. Registers are used for business planning, analysis, and evaluation as well as for legalisation and accounting of business entities. So the knowledge of general business functions and activities should be used to identify registered units. It means that the system analysis must start from the gathering of knowledge on the business planning, analysis, and evaluation procedures and on the legalisation and accounting of the institutional resources. It should be noted that formal registration procedures are usually established in the institution only in thecases required by the law. The register-based approach requires to apply formal unified registration procedures to all the units to be registered.

Task 2: *Determine which properties of registered units are basic, descriptive, and application-oriented properties.* The objective of this task is to determine which data must be registered necessarily and identify cross-references between the registers. Both the structure of the institution and institution business activities must be considered for this aim and the properties of global interest for each kind of registered units must be determined. External, for example, national registers must be analysed,

too, because some institution registers may be implemented as the local or application-oriented databases of external registers.

Task 3: *Develop registration standards.* The objective of this task is to develop the registration standards that serve as a basis to prepare institutional acts for establishing registers. The IRS should operate as a whole, integrity constraints must be maintained to the whole system. So the standardisation of registration rules and procedures as well as of register supervision and maintaining procedures is most preferable. To do this, the institutional acts on the registration must be prepared.

Task 4: *Identify the ground clusters.* The objective of this task is to identify the ground clusters of registers, and, for each ground cluster, to elaborate integration requirements. Our hypothesis is that ground clusters mirror essential institutional business activities and, to identify the ground clusters, those activities should be analysed.

Task 5: *Select a ground cluster and implement it.* The objective of this task is to select some ground cluster, to propose an architectural solution, to prepare requirement specifications for each register that belongs to this ground cluster, and begin designing and implementing the IRS. At the same time, we can begin specifying, designing, and implementing problem-oriented and task-oriented IS subsystems that are based on already implemented registers. It should be noted that the register requirement specification should include legal and organisational requirements as well as software, database, and security requirements. In our opinion, legal, organisational, database, and software requirements should be integrated on the basis of the notion of registration. Legal requirements specify kinds of registered units, classifiers and codifiers, registered data, registration documents, registration procedures, registration approval procedures, removal from the register procedures, obligatory services and service procedures. Organisational requirements specify register supervisors, overseers, and registrars, their rights and duties. Software requirements specify functional and non-functional requirements to data input, validation, transfer, browsing, auditing and other software programs included in the register system. Database requirements specify which data bases should be maintained in the register system.

Task 6: *Extend the ground cluster.* The objective of this task is a step-by-step extension of the selected ground cluster, implementing all registers that refer to it. Simultaneously, appropriate IS subsystems may be extended, too.

Tasks 5 and 6 must be repeated until all the ground clusters are implemented.

It should be noted that the proposed approach provides the activities and tasks that should be performed to develop the IS of an organization. In the case where the goal

is to develop a national or territorial register system, this approach must be slightly modified.

7. Conclusions

The registers play an important role in IS engineering. The register-based approach facilitates the integration of information systems on the national level. It enables us to reuse the data collected in the higher-level registers (up to the national level) and allows us to share most important data from the business point of view between applications. The proposed architecture of the register system increases the cohesion of IS subsystems, since data input, modification, and auditing procedures are transferred from the IS subsystems to registers themselves. It decreases the coupling between the databases, since the descriptive and the application-oriented data are separated. In addition, the register-based approach provides precise rules how to start a domain analysis, which entities to analyse first, and how to implement the system step-by-step. Legal, organisational, database, and software requirements may be integrated on the basis of the notion of registration. Institutional acts on the registration establish standards for services provided by registers, database maintenance procedures, and business data auditing and authorisation procedures. On the other hand, the costs to provide those standards are often relatively high due to the needed reorganisation efforts. The systematisation of business data facilitates to violate the privacy or institutional policy by receiving exhaustive data on the staff, business activities, and other confidential issues.

8. Acknowledgements

This paper intègrates the results of research achieved in two projects: elaboration of the regulations for Lithuania's state register system and the development of the Vilnius Gediminas Technical University IS. The authors thank Ona Bernotienė, Ona Jakštaitė, Algis Kupčinskas, and, especially Almantas Buitkus, for many fruitful discussions, suggestions, and comments on this work.

9. References

1. Jensen, P. Systematic statistical use of public data registers. In Kraus, H. (ed.) *The impact of new technologies on information systems in public administration in the 80's*. Proceedings of the first international IFIP conference on governmental and municipal data processing, Vienna, Austria, 23-25 February, 1983. Elsevier Science Publishers B.V (North-Holland), IFIP, 1983, pp. 303-328.
2. Aurbakken, E. Registers and data banks in a national administrative information system. In Bubenko J.A. jr., Čaplinskas, A., Grundspenskis, J., Haav, H.-M., Solvberg, A. (eds.). *Proceedings of the Baltic Workshop on National*

Infrastructure Databases: Problems, Methods, Experiences, Mokslo Aidai, Vilnius, 1994, Vol. 2, pp. 53-64.

3. Metra, I., Ogrins, A., Strodahs, O., Zitars, I. Some aspects of development of civil registration system in Latvia. In Bubenko J.A. jr., Čaplinskas, A., Grundspenskis, J., Haav, H.-M., Solvberg, A. (eds.). *Proceedings of the Baltic Workshop on National Infrastructure Databases: Problems, Methods, Experiences*, Mokslo Aidai, Vilnius, 1994, Vol. 2, pp. 65-68.
4. Buitkus, A., Čaplinskas, A., Markevičius, R. Lithuania 2000: strategy for Lithuanian national information infrastructure - approach and specific features. *Informatica*, Vol. 5, No. 3-4, 1994, pp. 283- 296.
5. Čaplinskas, A., Lupeikienė, A., Vasilecas, O. Integrated university information system as part of national information infrastructure. In Haav, H.-M., Thalheim, B. (eds.). *Databases and Information Systems, Proceedings of the Second International Baltic Workshop*, Tallin, June 12-14, 1996, Vol. 2, 125-135.