ACTA UNIVERSITATIS AGRICULTURAE ET SILVICULTURAE MENDELIANAE BRUNENSIS

Volume LVIII 1 Number 3, 2010

A DATA ORIENTED FRAMEWORK FOR DEVELOPING FLEXIBLE INFORMATION SYSTEMS

F. Dařena

Received: January 4, 2010

Abstract

DAŘENA, F.: *A data oriented framework for developing flexible information systems.* Acta univ. agric. et silvic. Mendel. Brun., 2010, LVIII, No. 3, pp. 13–20

As a consequence of a rapidly changing environment, the success of organizations is dependent upon the ongoing and immediate adjustments of their information systems as reactions to these changes. Therefore, flexibility becomes one of the most crucial features in information systems. This paper specifies a data model oriented framework for the development of flexible information systems. The result of such process is a system that is sufficiently general and flexible in relation to solving problems related to changing environment. From general requirements related to data layer of information systems the paper discusses the definition of major elements of logical data model (entities and their hierarchical arrangement, attributes of the entities and their important characteristics, relationships among entities and their characteristics) as well as possibilities of implementation and definition of application logic and data presentation. Proposed framework enables the organization to specify its own database structure, which best matches the situation of the organization and its environment. Because an approach similar to meta-data approaches is applied, methods for information sharing and interchange can be easily specified as well as program logic for manipulation with the data base on the application layer and data presentation.

data base, meta data, data analysis, data model, development framework

In order to provide managers and other employees with information, database and its content is crucial in any field. Specific examples are marketing related fields (Customer Relationship Management, Supply Chain Management), that are oriented not only towards the organization but also to its environment. Because these fields require information concerning the present, past and future they can be considered very information intensive (Dařena, 2007). Due to the dynamics of the environment and continuous organizational adaptation, the architecture of the information must adapt the change in order to follow the speed of business (Namba, Iijima; 2003). Such adaptation must also often reflect in the structure of organization's database. A database must therefore be flexible enough to facilitate these changes. Such flexibility also includes the ability of communication or information sharing with other business information systems or applications, e.g. through XML as a universal format (see e.g. Seligman, Rosenthal; 2001).

According to Tozer (1999), business data must meet the following criteria – accuracy, timeliness, consistency, transparency of meaning, and availability. Today, as a consequence of a rapidly changing environment, another criterion is becoming more important – the previously mentioned flexibility (Paschke, Molla, Martin; 2008). It's because the reaction of the organization is dependent upon the ongoing and immediate adjustment of the information infrastructure.

Systems integration, both internal and external, is a current important corporate issue. External systems integration refers to IT-mediated transactions between independent business entities, for example, between a company and its customers, suppliers, or other business partners, such as co-producers and banks (Lynne et al., 2003). Systems integration also plays a crucial role in various management approaches – Customer Relationship Management or Supply Chain Management – and must be part of the top management strategy (Salo, Karjaluoto; 2002). Many market subjects, business applications, as well as many approaches to systems integration require flexibility on the data layer (data integration is another tool for other integration approaches).

Information systems development and systems integration is a process that typically lasts relatively long and requires the cooperation of both the organization and information system's provider. However, the need for changes can occur relatively quickly and unexpectedly which calls for immediate adjustments to the organization as well as to the information system (information structure and content). Only in cases when the changes in the environment have a smaller impact the organizations can be flexible enough to be competitive (Wedemeuer, 1999). For this reason, it is advantageous when such adjustments can be performed by the organization itself. The advantages of this approach are also applicable to the side of IS provider – it is no longer necessary to keep multiple versions of an information system for different organizations which can be time and money intensive for all parties.

OBJECTIVE

It is effective to start with designing information infrastructure that has a common functionality before other IT planning steps, including integration of heterogeneous applications (Namba, Iijima; 2003). The paper focuses on definition of a framework that enables development of flexible information systems. The flexibility is aimed primarily to the data layer because of the reasons identified in the introductory section of the paper. The structure of a data model that is sufficiently general and flexible enough to solve the above mentioned problems is proposed. Thus, the ability to specify organization's own database structure that best matches the conditions in the organization and its environment is provided.

The length of the paper doesn't allow discussing also other issues related to other layers (application, presentation). However, because the approach that has some characteristics of meta-data approach is applied, methods for information sharing, interchange, program logic for manipulation with the database, and presentation processes can be easily specified and subsequently implemented.

METHODS

A solution similar to the one proposed in this paper was implemented, tested and operated in the University information system of Mendel University in Brno in the Czech Republic. This system is now in the phase of system operation (or implementation) at five universities with almost 60,000 users. The ability to modify the database structure (data model) enables modifications to be made to the structure of information that is stored pertaining various topics (students, employees, courses etc.) in various universities and university faculties or departments. In the early stages of the system operation, several weaknesses occurred. The most serious was the inflexibility regarding the definition of new data entities and their attributes that required interventions into database and sometimes into the core of the information system applications. For more details about this approach see e.g. Dařena (2002).

The framework proposed in this paper should solve the forementioned problems identified during the operation of the system – removing as much work as possible from the system administrators (database managers, programmers) and make the work more flexible through a higher level of abstraction and generality. The ability to trace selected values through time is also added.

Generally, the data meta model is based on the basic building blocks of systems analysis (Tozer, 1999):

- entities – objects, processes etc. that are in the scope of organizational interest,
- attributes – object properties,
- classification – defined relations between entities in terms of superiority and subordination,
- relationships – describe the associations among entities,
- domains – define allowed values for attributes.

If we leave aside obsolete approaches to database management (hierarchical and network model) we can choose from several modern database types (Blaha, 2001):

- enity-relationship model (RDBMS) – well developed, reliable, on the other hand, they provide a relatively small number of data types,
- multidimensional model (MDBMS) – faster than RDBMS for multidimensional analyses, better combines the facts and their dimensions,
- object models – relatively new paradigm, enables working with the same data types (more than in RDBMS) on both data and application layers, disadvantage can be seen in relatively small amount of theoretical foundations,
- object-relational model – combines the advantages of object and relational databases.

The creation of data meta model also results from the general process of creating a data model – see e.g. Von Halle (2002):

- identifying candidate entities and capturing meta data regarding each entity,
- grouping entities, looking for possibilities to denote subtype-supertype structures,
- determining relationships among entities, uncovering relationship cardinality for each relationship, considering relationship optionality, capturing meta data about relationships,
- identifying primary and alternate keys for each entity,

- replacing many to many relations by intersection entity,
- adding attributes, setting rules for attribute values – uniqueness, optionality, value checking etc.

Proposed framework should be implementable under any of above mentioned models easily and should allow storing all parts of data meta model. The proposed model should be also able to describe all business data which can be divided into the following categories (Basl, 2002):

- code lists – area codes, departments, banks, document types etc.,
- essential data – products, suppliers, customers etc.,
- operational data – related to business operation and essential data,
- archive data – past data,
- parameters – for the modification of information system operation (calculations, user interfaces etc.).

Existing meta data approaches

Most common application of meta data approach includes description of digital resources, most frequently web resources (Sicilia, 2006). Proposed framework is not oriented only to specific type of data and is therefore more general. Some of the meta data initiatives are focused only on one domain, e.g. IEEE LOM (see Malo, 2007), whereas others have general utilization. Discussed framework belongs to the latter group, its utilization should be general. Some models have only limited scope and can be used for description of limited amount of information. E.g. Dublin Core metadata model contains only a defined set of properties that can be used for resource description (ISO, 2009). Although the general concepts of the approach (see Powell et al., 2007) can be applied, all aspects of data modeling in information systems would not be covered. Other frameworks are based on particular technology for its implementation (e.g. RDF requires XML, see http://w3.org/RDF) which reduces the universality of such technologies.

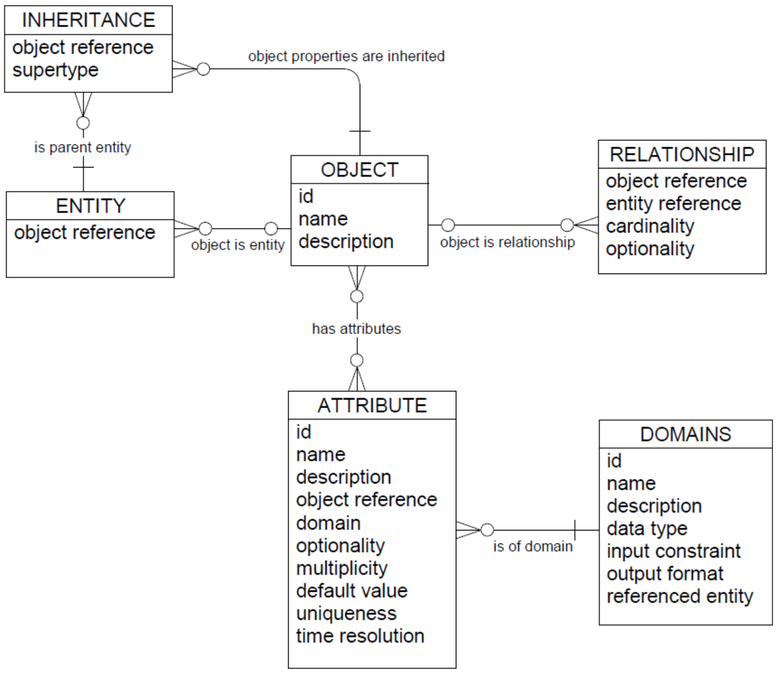
RESULTS

In this part, most of the attention is aimed at the most important of proposed framework – the data model. All other aspects that relate to application logic, presentation layer and implementation issues are mentioned just briefly.

Data model specification

The main elements of proposed logical data model represent the entities (can be also referred as objects classes) and relationships among them. In order to be able to work with all these elements equally in the entire information system it is necessary to define a data description set that is common for them. Relationships are, in principle, realized by the intersection entities with references to the entities participating in the relationship. Relations thus the have character of the entities. Meta information about entities and relationships (their basic properties) can be therefore stored in one way.

In order to enable multiple inheritance for entities (one entity can inherit attributes from more that one entity) the hierarchy of entities must be realized by many-to-many relationship (represented by Inheritance class in Fig. 1).



1: *Meta model of presented logical data model*

Detailed properties of relationships include familiar properties, such as the cardinality (on the side of the referred entity) and optionality (if the relationship is mandatory/optional for the referred entities). The number of occurrences of particular relationship defines its degree (number of entities that participate in the relationship). Thanks to this approach, relationships with any degree and also any cardinality (one-to-one, one-to-many, many-to-many) can be described.

Both entities and relationships can have several attributes. Because entities and relationships are realized by the same manner and the meaning of the attributes is the same in both cases, the implementation can be the same for both types of objects. The attributes definition includes reference to an object (entity or relationship), reference to attribute type (domain), optionality – whether the attribute is mandatory/optional for the entity/relationship, multiplicity – whether there can be more attribute values for one attribute, default value – implicit value for the attribute, uniqueness – whether the value must be distinct for all entities/relationships of the same class, time resolution – logical value, stating if we want to trace the attribute value through time. Multiplicity can be defined as the maximum number of attribute occurrences within one entity or relationship. This information can be entered as an integer number N – one entity/relationship can have maximum N values of the same attribute (e.g. maximum 5 phone numbers for one customer). If there can be an unlimited number of values for one attribute, 0 is chosen – an attribute with no (zero) occurrence doesn't make sense. Attribute type is again described by meta-data for the purpose of further manipulation within attribute values. Definition of an attribute type includes name, description, data type – basic data type of the attribute, input constraints – specifying attribute domain in detail (within the scope of data type), output format – describing the way of attribute values presentation, referenced entity – whether the attribute is a reference to an entity (i.e. values of this attribute are limited by existing values of some entity type).

Basic (primitive) data types include integer numbers in different ranges – positive numbers, size 1 B, 2 B etc., real (floating point) numbers in different ranges, fractions, dates, times, timestamps, strings, logical values, binary data, references to entities, functions (formula).

Composite attributes can be realized by using attributes with multiple values (instead of an array or list data types) or by using relationships with another entity that describes the structure of each attribute (instead of data type record). Binary data are described by their MIME type. Attributes of the type of Reference to an entity are used to optimize simple one-to-one or one-to-many relationships. Many-to-many relationships are always realized by the intersection entity with the possibility of specifying further relationship properties – see e.g. Shepherd (1990).

Functions (formulas) are expressions consisting of operators and operands. Operands include constants, values of the attributes of the entity itself or other entities and other function calls.

Input format defines constraints on the data when entered to the system. When the information concerning the input format is stored in the database it can be used at each stage of manipulation with the value. Because of separation from the application layer, the entire approach to given data manipulation is more flexible.

To constrain the values, a simple expression can be used that contains operands (attribute value represented by a variable, constants), operators (relational, logical, arithmetic), functions (mathematical, statistical, date and time, text, attribute – functions returning attribute values of various entities). To manipulate the strings, an instrument called "regular expressions" can be used. For more details see e.g. (Dařena, 2005).

Implementation

The design of the proposed data model doesn't require a particular way of implementation, it can be realized in relational DBMS, object DBMS, XML format etc. How individual parts of data model can be implemented is shown in Table I (based on Common Warehouse Metamodel (CWM) Specification, Version 1.0 from February 2001 (OMG, 2001).

I: *Implementation of entities, relationships and attributes in different data models*

Data source	Entity/Class	Attribute
object model	class	attribute
relational model	table	column
records	record	definition field
multidimensional model	dimension	dimensioned object
XML	element	attribute

On the application layer, different types of data sources should be hidden so the particular physical data model that is used is not important. However, the data model should be separated from the application logic in order to retain an information system that is flexible for making extensions and modifications (Tozer, 1999).

Demonstration of main benefits and limitations

During the development of Faculty information system a relational data model reflecting the current state of data structure was developed and implemented in a relational database. Individual data entities were on a regular basis implemented using tables, attributes represented by table columns. A major problem occurred at the moment of designing the information system for entire university (with the potential of offering this product to third parties). Attributes of the same entities on different faculties were often different which would lead to the situation where entity instances would be represented by sparse vectors (many empty columns) resulting in inefficient usage of data storage. With the increasing number of subjects related to the IS (faculties) the number of requests for adding/modification/deleting entities or their attributes was growing very fast.

Meeting such requirements required adding a database table or adding a column to a database table, and modification of all applications where this new piece information was needed (creating entities, modification of their attributes, presentation of values of such entities). Without moving to a system with higher flexibility the maintainability of the systems would decrease dramatically.

Implementation of the meta data concepts has brought most of the responsibilities to the hands of systems integrators. In the case a new attribute was needed for an entity for both internal and external applications (e.g. office hours and bank account number for employees, category of a person for meal ordering system) the modification of data model was in most cases made without the intervention of systems developers or administrators because fields in editing applications, fields in various reports, criteria for sorting or filtering data and others were generated automatically in related applications. This solution creates also a good background for systems based service oriented architecture (SOA) where the services represent the application logic (Erl, 2005) and can simply process data stored outside them. Generating data that are used as input for those services and storing data that are collected as the output from those services is a relatively easy task.

Main benefits of proposed framework:

- rapid speed of development of simple applications for editing simple dimension tables (e.g. programs of study, class rooms, courses), applications for creating new or modification of existing entities and their attributes are automatically generated,
- quick reactions to changed requirements regarding data structure (stored data items, their compulsoriness, acceptable values etc.),
- manipulating different objects in the same way, which leads to code reusability on one hand and increased user comfort on the other hand,
- automatic generation of on-screen or printed reports according user requirements,
- automatic generation of user defined documents for data interchange, facilitating integration of heterogeneous systems and enabling incorporating e.g. service oriented architecture principles into systems design,
- possibility of user defined handling of selected data types,
- easy conversion to different data models,
- the possibility of providing and generating information about database schema.

Insufficiencies of proposed framework:

- the performance of a database system is lower than in the situation when traditional approach is used,
- physical implementation and application of tools typical for database system selected for implementation cannot be defined or optimized directly through this framework,
- proposed framework is not suitable for entities with large number of instances, where the data is accessed very often because of inefficient data processing and manipulation,
- management of user access privileges cannot be automated by presented tools and must be implemented separately (in applications or e.g. by procedural extension of a database system),
- the best features provided by various database systems are not used to the full extent.

CONCLUSION

Information systems are often being rebuilt in order to implement all necessary functionality. However, single solutions rarely cover all business needs and the information system must be therefore extended by another applications inside as well as outside the organization (Namba, Iijima; 2003) or the applications must be modified. This paper describes a framework for developing flexible information systems. The main contribution is the definition of a novel approach to data modeling that forms the essence of entire framework. Proposed data model is suitable for developing applications that depend more on information rather than on complex calculations and where flexibility plays crucial role. Applications of this type include Enterprise Resource Planning, Customer Relationship Management, Decision Support Systems, Expert Systems, marketing applications and others. Making an information system flexible in the way, that new objects of interest including their attributes and relationships can be added enables reducing the semantic distance by enabling users to better describe their problems and to directly manipulate objects related to the problems (Hutchins, Hollan, Norman; 1985).

Previous version of proposed model (see Dařena, 2002) was successfully implemented by University information system that is being developed at the Mendel University in Brno and is operated at three other universities in the Czech Republic and abroad. Experience with the development of the infor-

mation system shows that changes of user requirements and changes in the organization's environment that demand changes in the information system occur quite often. This can lead to poor and difficult maintenance of the information system. This paper presents improved version of the framework that is able to face many problems that arose in former implementation.
The approach specifying the data model makes the entire approach flexible and independent of the particular implementation (programming language, physical data model, database management system, platform, communication technologies etc.). The program logic describing the manipulation with the information (inserting, updating, deleting, presentation) can be generally implemented for all entities and relationships and their attributes. Programming special applications to manipulate individual types of information can be avoided (such applications are “generated” automatically) or such applications can be developed very easily.
The approach also enables data for various kinds of applications that are either part of the information system or isolated applications to be made more readily available. Such applications include e.g. Business Intelligence analytical tools (OLAP and Data Mining techniques), expert systems (data for the knowledge base of an expert system), case based reasoning systems (data create the case base), decision support systems (data for decision models) etc. Thanks to the proposed structure, conversion e.g. to XML format is straightforward, comprehensible and quick. All this facilitates communication and the process of systems integration.

SOUHRN

Datově orientovaný rámec pro tvorbu flexibilních informačních systémů

V důsledku rychle se měnícího prostředí jsou reakce organizace závislé na průběžném a bezprostředním přizpůsobování se těmto změnám. Tato situace si vynucuje rovněž přizpůsobování v oblasti podpory podnikových procesů pomocí informačních systémů. Flexibilita se tedy stává jednou z významných vlastností informačních systémů. Tato práce specifikuje datově orientovaný rámec pro vývoj flexibilních podnikových informačních systémů. Výsledkem tohoto procesu je systém dostatečně obecný a flexibilní, který je schopen čelit problémům spojeným se změnami v prostředí, v němž organizace působí. Od obecných požadavků na datovou vrstvu informačního systému je postupováno přes obecnou definici hlavních elementů logického datového modelu (entity a jejich hierarchické uspořádání, atributy entit včetně jejich významných vlastností, vztahy mezi entitami a jejich vlastnosti, datové typy a obory hodnot jednotlivých charakteristik uložených v databázi) až po stručné nastínění možností implementace a definice aplikační logiky a prezentace dat. Navržený rámec umožňuje organizaci a uživatelům podnikového informačního systému definovat vlastní databázovou strukturu, která nejlépe popisuje stav organizace a jejího okolí. Protože je zvolen postup vycházející z metadatového přístupu, jsou poměrně snadno technicky realizovatelné nástroje a metody pro ukládání, sdílení a přenos informací, stejně jako aplikační logika pro manipulaci s daty na aplikační úrovni.

databáze, metadata, datová analýza, datový model, vývojový framework

Acknowledgements

This paper is supported by the Research program of Czech Ministry of Education number VZ MSM 6215648904/03/03/05.

REFERENCES

BASL, J., 2002: Podnikové informační systémy. Praha: Grada, 142 p. ISBN 80-247-0214-2.

BLAHA, M. R., 2001: A Managers Guide to Database Technology. Harlow: Prentice Hall, 260 p. ISBN 0130304182.

DAŘENA, F., 2002: University information system's kernel. Master thesis, Mendel University Brno.

DAŘENA, F. 2005: Myslíme v jazyku PERL. 1. vyd. Praha: Grada, 700 s. ISBN 80-247-1147-8.

DAŘENA, F., 2007: Global architecture of marketing information systems. Agricultural economics: Zemědělská ekonomika. 52 (9): 432–440. ISSN 0139-570X.

ERL, T., 2005: Service-Oriented Architecture. Upper Saddle River: Pearson Education, 760 p. ISBN 0-13-185858-0.

HUTCHINS, E. L., HOLLAN, J. D., NORMAN, D. A., 1985: Direct Manipulation Interfaces. Human-Computer Interaction, 1: 311–338.

ISO 15836:2009 Information and documentation – The Dublin Core metadata element set. ISO, 2009.

LYNNE, M. M., AXLINE, S., EDBERG, D., PETRIE, D., 2003: The Future of Enterprise Integration: Strategic and Technical Issues in Systems Integration. Competing in the Information Age: Align in the Sand. New York: Oxford University Press: 252–287, ISBN 0195159535.

MALO, R., 2007: Implementace standardů v eLearningových systémech. Acta Universitatis Agriculturae et Silviculturae Mendelianae Brunensis, LV (3): 161–170.

NABMA, Y, IIJIMA, J., 2003: EII Meta-model on integration framework for viable enterprise systems. Journal of systems science and systems engineering, 12 (1): 111–126.

PASCHKE, J., MOLLA, A., MARTIN, B., 2008: The Extent of IT-Enabled Organizational Flexibility: An Exploratory Study among Australian Organizations. In: Proceedings of 19th Australasian Conference on Information Systems.

POWELL, A., NILSSON, M., NAEVE, A., JOHNSTON, P. BAKER, T. DCMI Abstract Model. 2007. http://dublincore.org/documents/abstract-model/ [cit. 2010-02-25]

SALO, J., KARJALUOTO, H., 2002: IT-Enabled Supply Chain Management. Contemporary Management Research, 2, 1: 17–30. ISSN 1813-5498.

SELIGMAN, L., ROSENTHAL, A., 2001: The impact of XML on Databases and Data Sharing. IEEE Computer, June 2001.

SHEPHERD, J. C., 1990: Database Management: Theory and Application. Homewood: Irwin, 793 p. ISBN 0-256-07829-7.

SICILIA, M. A., 2006: Metadata, semantics, and ontology: providing meaning to information resources. International Journal of Metadata, Semantics and Ontologies., 1 (1): 83–86.

TOZER, G., 1999: Metadata Management for Information Control and Business Success. Boston: Artech House, 318 p. ISBN 0890062803.

VON HALLE, B., 2002: Business Rules Applied. New York: John Wiley & Sons, 592 p. ISBN 0471412937.

WEDEMEIJER, L. 1999: Design the flexibility, maintain the stability of Conceptual Schemas. Lecture notes in computer science. Berlin: Springer. 467–471. ISSN 0302-9743.

Address

Ing. František Dařena, Ph.D., Ústav informatiky, Mendelova univerzita v Brně, Zemědělská 1, 613 00 Brno, Česká republika, e-mail: frantisek.darena@mendelu.cz