Economic Insights – Trends and Challenges	Vol.III (LXVI) No. 3/2014	91 - 98

Document Management Processes and Use Case Scenarios Elaboration

Aurelia Pătraşcu

Faculty of Economic Sciences, Petroleum-Gas University of Ploieşti, Bd. Bucureşti 39, 100680, Ploieşti, Romania
e-mail: patrascuaura@yahoo.com

Abstract

This paper aims to present the objectives and the stages achieved in order to model a document management system, using Visual Paradigm for UML as a modelling tool. An information system modelling for document management domain using UML language provides different perspectives that guide developers in using object concepts, offers a graphical notation, behind which an adequate semantics is found and supports building and documenting the components of document management software system.

Keywords: *document record, document management system, use case, static modelling, UML*

JEL Classification: *C81, C61*

Introduction

According to the in force legislation (Arhivele Naţionale 1996; Arhivele Naţionale 2001; Law No. 16 1996; Law No. 358 2002; Law No. 474 2006), the organizations are obliged to record at the general registrar's office in a single input-output register or in more registers all the documents that enter in the organization, exit from the organization or are elaborated in order to be internally used. The registration numbers assigned to documents must not be repeated, if the organization generates a unique documents number.

The activity of documents record indicates, in fact, the documents official existence and signifies their birth certificate. The insurance of preserving them depends, widely, on the record correctness. Due to the large amount of documents that are received or sent by an issuer, the institution must ensure an accurate organization of their circuit.

The documents record is achieved by each work department, if the organization receives issues and elaborates a large amount of documents that will be internally used. Also, their registration number assigned by the sender and the department name where the documents are distributed for recording and resolution are noted at the general registrar's office.

The documents are recorded in chronological order, depending on their receiving date, starting from January 1st to December 31st of every year. The input-output register structure is presented in Figure 1.

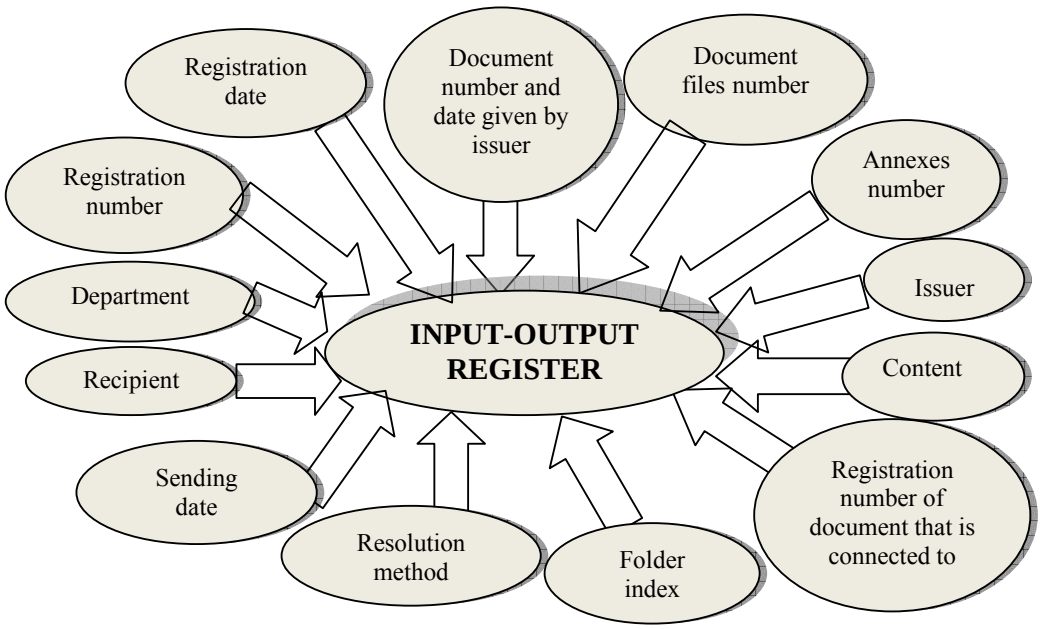

Fig. 1. Input-output register structure.

Source: made by the author

The registrar's office most important operation consists in receiving, recording and distributing documents in order to resolve them. Each document will be assigned to a folder, depending on its content.

Each issuer elaborates a folders classification in a table form (Table 1) for its own documents. In this classification that is sorted by the work departments are recorded the documents types group by problems and storage terms. This represents an analytical element and is used both by registrar's office in order to allocate the entries to departments for resolution and, especially, by departments in order to assigned documents to folders after resolution, because the folder will have this significance only when is resolved and completed.

In accordance with in force legislation, the folders classification is the creation tool for the current archive. All the subsequent activities are depending on its existence and on its correct application, the activities being represented by: documents classification by problems and storage terms, archival units' creation, documents inventory, documents selection, documentary information capitalization.

Table 1. Archival classification structure.

Column Number	Content	Explanations
1	work departments names	are noted in the order given by the issuer organizing scheme and are numbered using roman numerals
2	work departments subdivisions	are numbered with capital letters
3	content of the documents that form the folder	is numbered with Arabic numerals, starting with number 1, at every work department
4	storage term	is established taking into account the in force legislation, the practical significance for the documents issuer activity and, particularly, the scientific significance of the information that are contained by documents.

Source: Duşmănescu, Pătraşcu and Tănăsescu (2006), pp. 25; Arhivele Naţionale (2001), pp. 26.

Without the existence of a classification, an archive becomes a heterogeneous conglomerate of documents whose reorganization requires a documents reclassification and, thus, a duplication of the work. Therefore, documents creation based on the classification is the most economical and the most operative system (Berciu-Drăghicescu et al, 2003).

Use Case Scenarios Elaboration for Document Management

The modelling process represents the base process used by analysts in their effort to facilitate the understanding of the phenomena that occur in a system, in order to increase its efficiency and to improve its performances. This process consists in a range of stages that must be followed to convert the idea, firstly, in a conceptual model and, then, into a quantitative representation.

Unified Modelling Language (UML) has appeared in order to eliminate the limits and the multiple differences between symbols, notations or diagrams types existing in object-oriented methodologies. These aspects generated difficulties about the understanding, taking over and using them by different users groups, at new systems creation or in systems maintenance process (Dumitraşcu et al 2005; Fowler 2003; Lungu and Bâra 2007; Pătraşcu and Tănăsescu 2010; Pender 2003; Rumbaugh, Jacobson and Booch 1999; Tănăsescu 2014; Tănăsescu and Pătraşcu 2013; Udrică, Martinov and Lupoaie 2009).

Use case diagram is the use cases and actors graphical representation and is, often, built with a textual description. This diagram documents the interactions of the system with the environment, respectively, with the human factor, with other systems and between the computers (Lungu and Bâra 2007).

In a document management system that will be used by a City Hall, the actors are represented by:

- City Hall employees (application users – issuer, sender user, recipient user);
- City Hall clients (submit requests in document form – external user);
- application administrator (performs application current maintenance operations – system administrator);
- system actors (register, folder system and document system).

A *use case* represents a set of actions sequences executed by the system that generate a noticeable result, interesting for a certain actor (Dumitraşcu et al 2005).

The use case diagram presented in Figure 2 has been created using Visual Paradigm for UML modelling tool (Visual Paradigm 2014) and models a document management system from a public institution.

Use case diagram at document level (Figure 2) is composed of seven use cases: ***Verify Document Existence***, ***Create Document***, ***Access Pages***, ***Modify Document*** (***Modify Content***, ***Modify Characteristics***), ***Save New Version***, ***Archive Document*** and ***Delete Document*** that treat all possible situations in which the actors can interact with the document management system. This first diagram contains the following actors: the issuer, the sender user, the recipient user, the register and the folder system.

Create Document use case description is detailed in Table 2 and comprises the following information: use case name, primary actor, secondary actor, objective, preconditions, post-conditions, nominal scenario and alternative scenario (optional).

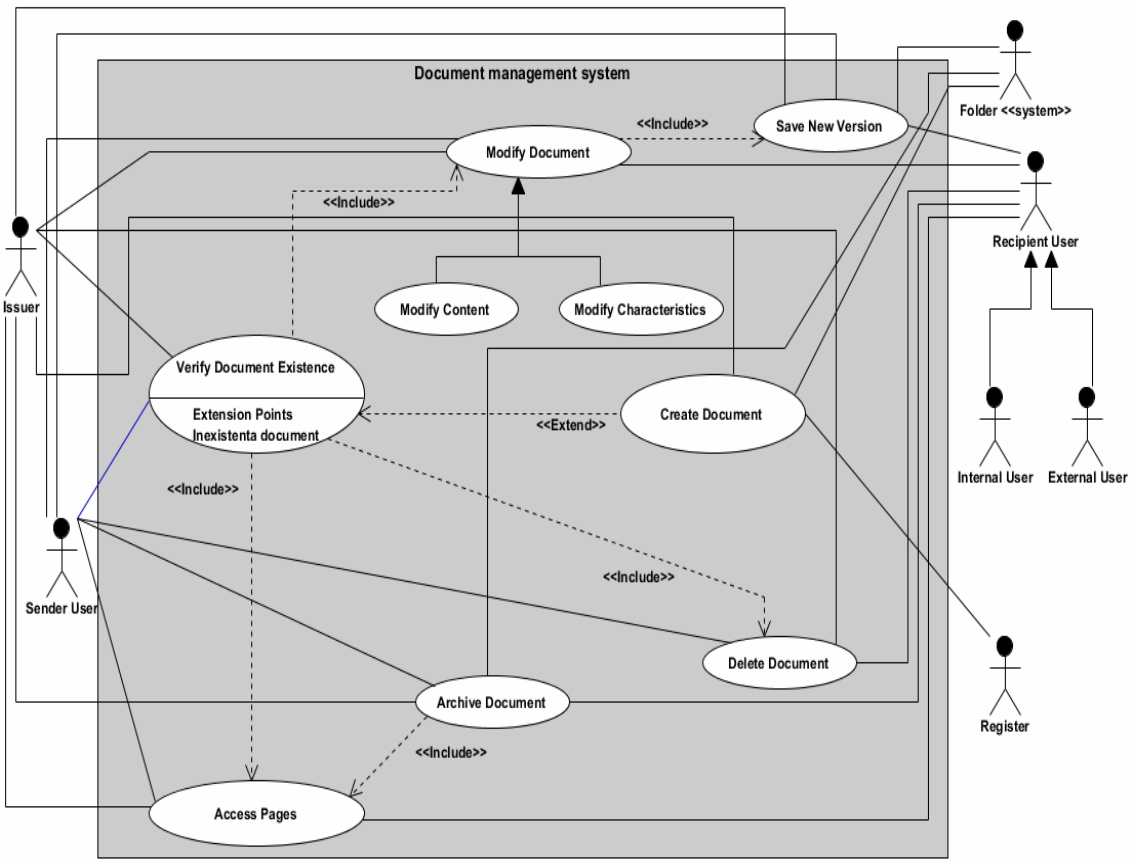

Fig. 2. Use case diagram at document level

Source: made by the author.

Table 2. Create Document use case detailed description

Use case name	***Create Document***
Primary actor	***Issuer***
Secondary actor	***Folder <<System>>, Register***
Objective	Document storage in electronic format into the database and easy retrieval using both a quick search by registration number and an advanced search by more criteria.
Preconditions	Document absence from the document management system
Post-conditions	The document is created by issuer and can be saved and stored into the database in order to be easily retrieved by the application users.
Nominal scenario	1. The issuer verifies the document type existence in the documents types' classification. 2. The system displays a message. 3. The issuer specifies the document registration number. 4. The system creates a record in the input-output register for this action. 5. The issuer indicates the document main characteristics: data the document enters into the system, comments, users that can access the document, versions, updates number and its restrictions (optional). 6. The issuer uploads the file (files) adequate to this document and selects the document entry type.

Source: made by the author.

Static Modelling for Document Management

Class diagram is a diagram type used in order to describe the static structure, namely *entities* or *classes* that exist in a system. This diagram type is, often, used by developers to specify classes, but can be, also, very useful for specifying the systems or subsystems structure from a real business (Dumitraşcu et al 2005; Fowler 2003; Pender 2003; Tănăsescu 2014).

Analysing the activity regarding document management, I have identified and defined 18 objects classes that are presented in figure 3. This diagram represents an improvement of a previous version (Pătraşcu and Tănăsescu 2008), where the classifications corresponding to the main concepts from document management domain were not distinctly emphasized. The archiving concept is, also, treated in this new version.

As it can be seen in figure 3, this diagram classes are: the existing documents (***Document***), the document pages (***Pages***), the entries classification (***Entries***), the folders that contain documents (***Folder***), the location where folders are found (***FolderLocation***), the registers where the folders are included in (***Register*)**, the classifications for documents, folders, registers, menus, metadescriptors (***DocumentClassification, RegisterClassification, FolderClassification, MenusClassification, MetadescClassification***), the departments that have issued documents (***Departments***), the function within the department (***Function***), the users (***User***), the user groups (***UsersGroups***) and the archives represented by ***FolderArchive, DocumentArchive, PagesArchive***.

A class diagram can contain both relationships between classes and relationships between classes' instances. Relationships between classes are represented by generalization relationships, dependency relationships and realization relationships. Relationships between classes' instances are the association relationships and aggregation /composition relationships.

In the document management application, I have defined the following relationships types:

- *4 composition relationships* between ***Document*** and ***Register*** classes, ***Folder*** and ***Register*** classes, ***Function*** and ***Departments*** classes, ***Pages*** and ***Document*** classes (e.g. an object of the document class is part of a register class instance and is not part of another instance);
- *8 "one to many" unidirectional association relationships* between ***RegisterClassification*** and ***Register*** classes, ***FolderClassification*** and ***Folder*** classes, ***DocumentClassification*** and ***Document*** classes, ***Folder*** and ***FolderLocation*** classes, ***Pages*** and ***Entries*** classes, ***User*** and ***Function*** classes***, Folder*** and ***User*** classes, ***Document*** and ***User*** classes;
- *3 "one to one" bidirectional association relationships* between ***Document*** and ***DocumentArchive*** classes***, Folder*** and ***FolderArchive*** classes, ***PagesArchive*** and ***Pages*** classes;
- *5 "many to many" bidirectional association relationships* between ***MenusClassification*** and ***UsersGroups*** classes, ***UsersGroups*** and ***User*** classes***, Folder*** and ***Document*** classes, ***Folder*** and ***MetadescClassification*** classes, ***Document*** and ***MetadescClassification*** classes.

Object diagrams, also known as instance diagrams, present objects and links between them (Cogzarea 2010; Dumitraşcu et al 2005; Tănăsescu 2014; Udrică, Martinov and Lupoaie 2009).

Like class diagrams, object diagrams represent static structure. Object diagrams are used, mainly, to present a context, for instance the situation before or after an interaction. However, they are useful because they support the understanding of complex data structures, such as recursive structures.

An object representation can, also, include attribute values. In figure 4 is illustrated an object belonging to ***Document*** class.

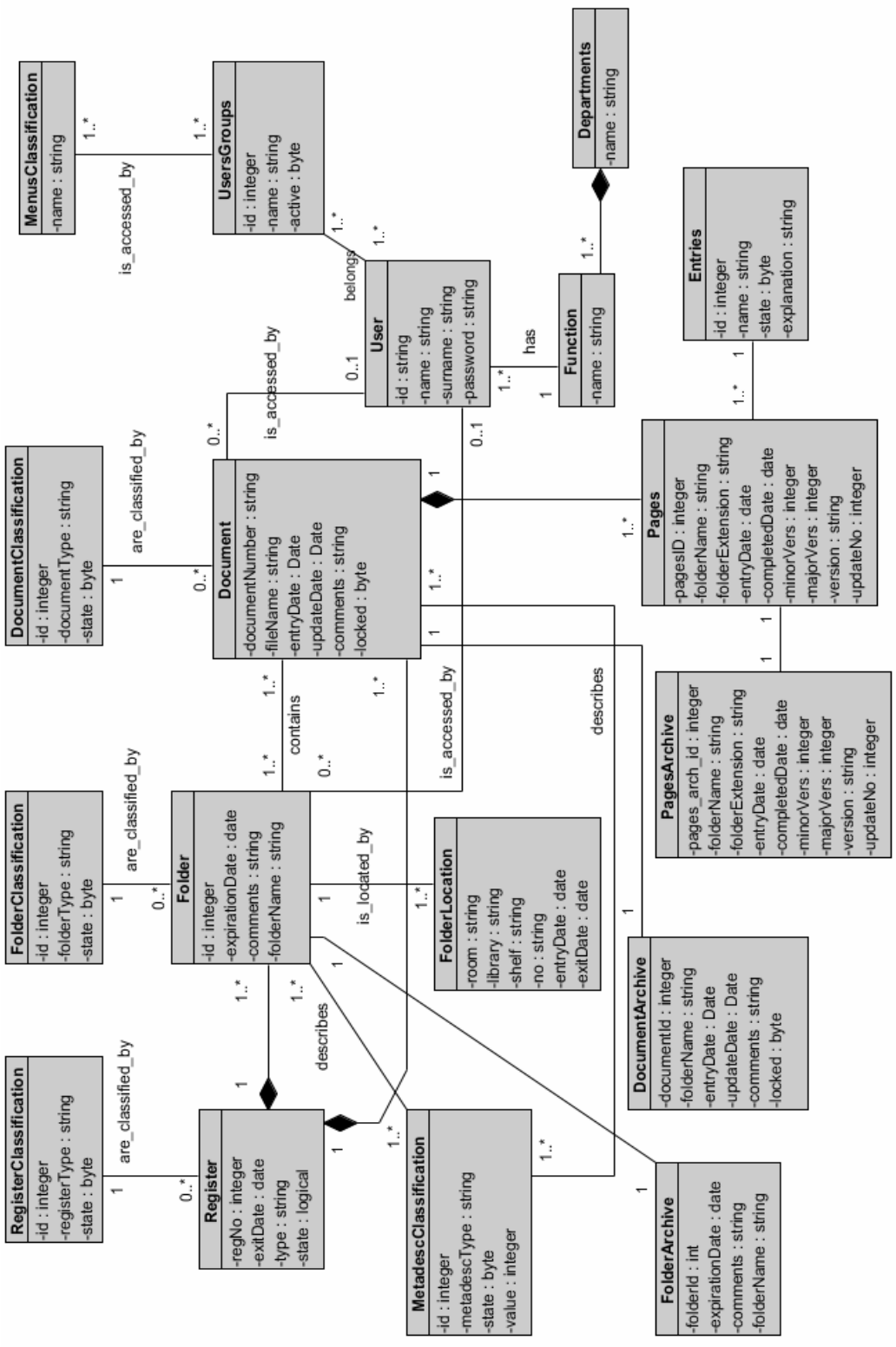

Fig. 3. Class diagram

Source: made by the author

Document1:Document
documentNumber=1967 fileName=”Invoice1” entryDate=01/07/2014 updateDate=- comments=- locked=0

Fig. 4. An instance of Document class

Source: made by the author

Any UML construction (use cases, classes, objects) can be grouped into higher level units as *packages*. In a UML model, each class belongs to a single package and has a distinctive name in the package. At the same time, packages can be members of other packages, leading to a hierarchical structure of packages and classes. A package can, also, contain sub-packages and independent classes. In UML, package name is followed by character "::" that separates it from the sub- package name or the component class name (Lungu and Bâra 2007; Pender 2003).

The package diagram of document management application is presented in Figure 5. The diagram central package is ***Documents*** package which is connected with the other application packages: ***Addresses, Users, Classifications, Workflow*** and ***Archives***.

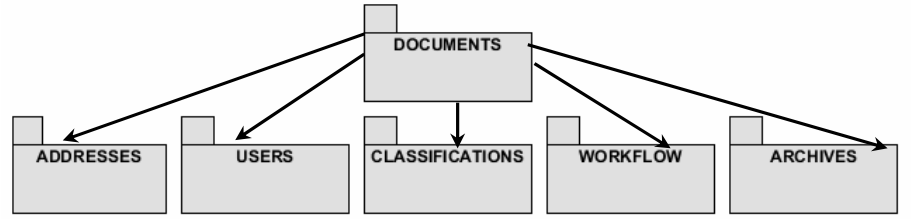

Fig. 5. Package diagram

Source: made by the author

Conclusions

Static models contain numerous entities for complex information systems. The large amount of classes can be assimilated and more easily understood by the development team members, if more class diagrams are built, grouping classes with certain common characteristics (e.g. a grouping criterion can be given by the classes types: boundary, entity and control).

The main goal of the document management system modelling is represented by the understating the system context and terms dictionary.

An information system modelling for document management domain using UML language provides different perspectives that guide developers in using object concepts, offers a graphical notation, behind which an adequate semantics is found and supports building and documenting the components of document management software system. It, also, allows systems specification through precise, unambiguous and complete methods at all detail levels: analysis, design and implementation.

UML models for document management can be used to model data on three levels: conceptual, logical and physical, according with the data abstraction principle.

The diagrams that are presented in this article represent a starting point in building a document management system for a public administration institution. These diagrams can be easily used by others designers because they have the following characteristics:

- symbols, diagrams types and models are standardized;
- contains elements that are explicitly represented;
- are flexible and easy to maintain;
- can be quickly modified in order to be used at the analysis and designing of other systems from the same activity domain.

References

1. ***, *Legea Arhivelor Naţionale nr. 16/1996*.
2. ***, *Legea nr. 358/2002 pentru modificarea şi completarea Legii Arhivelor Naţionale nr.16/1996*.
3. ***, *Legea nr.* 474/2006 *privind aprobarea Ordonanţei de urgenţă a Guvernului nr.39/2006 pentru modificarea şi completarea Legii Arhivelor Naţionale nr. 16/1996*.
4. Arhivele Naţionale, *Instrucţiunile privind activitatea de arhivă la creatorii şi deţinătorii de documente,* aprobate de conducerea Arhivelor Naţionale prin Ordinul de zi nr. 217 din 23 05.1996.
5. Arhivele Naţionale Direcţia Judeţeană Prahova, *Arhivele Prahovei. Legislaţie arhivistică*, Ploieşti, 2001.
6. Berciu-Drăghicescu, A. et al, *Manual de secretariat şi asistenţă managerială*, 2003, available at http://ebooks.unibuc.ro/StiinteADM/secretariat/index.htm [accessed on 18th August 2014].
7. Cogzarea, G., *Metodologii orientate pe obiecte utilizate în proiectarea sistemelor informatice*, Editura Infomega, Bucureşti, 2010, ISBN 978-973-7853-44-8.
8. Dumitraşcu, L. et al, *Analiza şi proiectarea orientată obiect a sistemelor informatice cu UML*, Editura Universităţii din Ploieşti, 2005, ISBN 973-719-012-2.
9. Duşmănescu, D., Pătraşcu, A., Tănăsescu, A., A Data Model for Documents' Electronic Archiving, *Revista Informatică Economică*, Editura INFOREC, Bucureşti, vol. 10, nr. 1, 2006, pp. 25-28, ISSN 1453-1305.
10. Fowler, M., *UML Distilled: A Brief Guide to the Standard Object Modeling Language (3rd Edition)*, Addison-Wesley Professional, 2003, ISBN-10 0321193687, ISBN-13 978-0321193681.
11. Lungu, I., Bâra, A., *Sisteme informatice executive*, Editura ASE, Bucureşti, 2007, ISBN 978-973-594-690-6.
12. Pătraşcu, A., Tănăsescu, A., A document management system modeling, International Scientific Conference "European Integration – New Challenges for the Romanian Economy, 4th edition, Oradea, May 30-31, 2008, *Analele Universităţii din Oradea, Seria Ştiinţe Economice*, Ediţie pe suport CD-ROM, TOM XVII, 2008, pp. 1479-1484, ISSN-1582-5450.
13. Pătraşcu, A., Tănăsescu A., Use Case Scenarios Developing for Document Management*, Proceedings of the 14th International Business Information Management Association Conference*, 23-24 June, Istanbul, Turkey, 2010, pp. 2490-2498, ISBN 978-0-9821489-3-8.
14. Pender, T., *UML Bible*, John Wiley & Sons, 2003, ISBN 978-0-7645-2604-6.
15. Rumbaugh, J., Jacobson, I., Booch, G., *The Unified Modeling Language Reference Manual*, Addison-Wesley, 1999, ISBN-10: 020130998X, ISBN-13 978-0201309980.
16. Tănăsescu A., Pătraşcu A., *Proiectarea sistemelor informatice*, Editura Universitară, Bucureşti, 2013, ISBN 978-606-591-612-8
17. Tănăsescu, A., Object oriented modelling, a modelling method of an economic organization activity, *Annals of the „Constantin Brâncuşi" University of Târgu Jiu, Economy Series*, Special Issue/2014, Information society and sustainable development, pp. 275-281, ISSN 2344 – 3685, ISSN-L 1844 -7007.
18. Udrică, M., Martinov, M.D., Lupoaie, A.E., *UML prin aplicaţii. Studii de caz privind dezvoltarea sistemelor informatice*, Editura Renaissance, Bucureşti, 2009, ISBN 978-973-8922-38-9.
19. http://www.visual-paradigm.com