A DOCUMENT MANAGEMENT SYSTEM MODELING

Pătrașcu Aurelia
Petroleum and Gas University of Ploiești, Economics Sciences Faculty, patrascuaura@yahoo.com

Tănăscu Ana
Petroleum and Gas University of Ploiești, Economics Sciences Faculty, atanasescuro@yahoo.com

The authors propose in this paper a model for a document management system, describing, briefly, the document management system and the model concepts, as well as, the Unified Modeling Language characteristics. Finally, they are presented some more important diagrams of this language.

Key words: Model, Document Management System, UML, Class Diagram, Use Case Diagram

Introduction

Business processes streamlining and productivity increasing are fundamental concerns for any organization (private, public or non-profit) [10]. In an increasingly strict regulatory environment, the document and records management is one of the major objectives of any organization.

The electronic document management has represented a continuous concern for computer scientists from all over the world and has developed as an important activity in the document archiving and retrieving field in the documentation centres. Although, almost all the documents from an organization are electronic documents, they can not be effectively used if they are not integrated into a management system. The present technologies, both hardware and software technologies, allow the presentation of the electronic document management concept in a new context, the collaboration and cooperation context in a knowledge based economy. The document management term is a part of a wider concept that covers the communication and knowledge management aspect of an organization [6].

The document management is an information system that allows the transfer, the storage and the retrieving of documents and records, potentially both paper and electronic, including workflow, content and knowledge management. The reality has proven that electronic documents and records are here to stay, at least for the foreseeable future. Every email is a record and every file is a source of information that carries a cost. Organizations avoid the challenge but they must accept the need to effectively manage their electronic documents [11].

![Fig. 1. Document management life cycle [9].](image-url)
The authors propose in this paper a model for a document management system (DMS). For that purpose, they are defined, briefly, the DMS and the model concepts, they are described the UML language characteristics and, finally, they are presented some more important diagrams of this language.

**UML characteristics**

The model is an abstract representation of a system that allows system performances planning, study, analysis, conception and control before its proper achievement, as well as, the design documentation creation, facilitating at the same time the communication between the design teams [3]. This model construction implies using of a modeling language, i.e. a communication mean for all the persons involved (designers and users) to understand the same thing.

Currently, the object modeling industrial standard is UML (Unified Modeling Language). UML is a fundamental graphic language that is defined round three diagrams categories, each of them being used in representation of some information system particular concepts. The first category describes the functional services, the second one represents the system static structure and the third one refers to the dynamics of the system running [4].

UML is the standard language for specifying, visualizing, constructing and documenting the artefacts of a software system [8].

**UML – a specifying language**

In the software engineering, the specification means to elaborate precise, complete and without ambiguities models. A visual specification, such as UML specification, is exposed to the risk of generating ambiguities in the communication process in a higher degree than the formal specification. However, a visual specification of the solution in all the software system development stages is preferred due to the importance of better communication between the software engineering specialists (users and beneficiaries) as partners in the same project [2].

**UML – a visualizing language**

UML emphasizes the visualizing of a software solution to remove the biggest part of the communication problems that can arise between different participants’ categories at a software development [2].

**UML – a constructing language**

The models elaborated in UML can be easily translated in numerous programming languages. This state of things suggests the natural mapping possibility of a model in languages like Java, C++, Visual Basic etc. [2].

**UML – a documenting language**

A real software company produces all kind of components additional to executable code obtaining of a software system like [2]:

- requirements regarding software system;
- system architecture describing;
- design specifications;
- source code;
- development plans etc.

The UML language version 1.0 uses ten types of diagrams (fig. 2) that can be, thus, classified [5].
The current version (UML 2.0) adds three more diagrams: composite structure diagram, interaction overview diagram and timing diagram.

The functional model is represented in UML by the use case diagrams that describe the system from the user point of view [7].

The object model is represented in UML by the class diagrams and object diagrams that describe the structure system in terms of classes, objects, attributes, associations and operations [7].

The dynamic model is represented in UML by the sequence diagrams, communication diagrams, state machine diagrams and activity diagrams that describe the system intern behaviour [7].

**System analysis and modeling**

For system modeling the authors have used the Visual Paradigm for Unified Modeling Language development environment. This environment was created by the Visual Paradigm Company for designing of some information system object-oriented models and for generating, finally, the adequate code in Java Language [3, 12]. In Visual Paradigm are available all the diagrams, both static diagrams and functional and dynamics diagrams, the development environment allowing their creation and maintenance in the smallest details.
The authors present, further, the Use Case Diagram and the Class Diagram regarding the DMS. The Use Case Diagram produces a first view upon the system structure, a design point of departure, an identification of the objects and sequence diagrams [3]. This diagram’s elements are: the actors (interns, externs, subsystems), the use cases and the relations between use cases.

The Use Case Diagram proposed by the authors is presented in figure 3.

In the case of DMS, the actors, i.e. the roles which the users are playing with the respect to the system, are: the system administrator, the database operator, the database and the document.

The use cases contains a set of actions sequences indicating what must be designed (conversion, document saving, deny saving).

The Class Diagram is the most important diagram in the object-oriented analysis and design. The diagram’s purpose is to structure the static nature of the classes in terms of attributes, operations and relationships [5].

This contains classes and relationships between classes. A class is a model for objects with similar structure, behaviour and relationships. Each class has a name, attributes and operations.
Analysing the activity regarding the document management, the authors have identified and defined 8 object classes: the existing documents (Document), the departments that issued the documents (Departament), the files in which are the documents are included (Dosar), the access restrictions (Restrictii_acces), the downloading or uploading mirrors (Ferestre_dw/upload), the access requests (Cereri), the redirecting departments (Dep_redirectionare), the operators (Operator_dw/upload) and 4 subclasses regarding the document type (Text, Film, Imagine, Sunet). There are, also, 2 special classes Conversie and Download/upload that are association classes (figure 4).

Fig. 4 The Class Diagram.

A Class Diagram can contain both relationships between classes and relationships between classes’ instances.

Relationships between classes are represented by generalization, dependency and realization. The diagram from this paper includes only generalization relationships. Generalization is a subclass-class relationship. According with the figure, in the diagram there are 4 generalization relationships between Document superclass and Text, Film, Imagine and Sunet subclasses (any instance of the Text, Film, Imagine and Sunet subclasses is, also, an instance of the Document class, inheriting the attributes and operations of this class).

Relationships between classes’ instances are represented by association, aggregation and composition. In the application are traced:

- 3 compositions between Document and Dosar, Dosar and Departament and Document and Departament classes (e.g., an object of the Document class is a part of an instance belonging to the Dosar class and it can’t be a part of other instance);
- 3 “one to one” bidirectional associations between Text, Film, Imagine and Sunet classes (e.g. a film document can be converted in one and only one text document and vice versa);
- 5 “one to many” unidirectional associations between Cereri and Dep_redirectionare, Cereri and Operator_dw/upload, Cereri and Operator_dw/upload, Document and Ferestre_dw/upload and Document and Restrictii_acces classes (e.g., a mirror can dispose of all of its requests while a request can’t know what mirror have used).
Conclusions
The authors have realized a model for DMS using UML language. The UML language has the following advantages:

- is easy to use by the users;
- ensures extensibility;
-formulates specifications independent of a certain programming language and system working processes.

References