U_VBOOM : UNIFIED ANALYSIS AND DESIGN PROCESS BASED ON THE VIEWPOINT CONCEPT

Abdellatif Hair
Faculty of Sciences and techniques P.O. Box. 523 BENI MELLAL MOROCCO

Keywords: Analysis/ Design process, modeling, Viewpoint, UML.

Abstract: The introduction of viewpoint in object-oriented design provides several improvements in modeling complex systems. In fact, it enables the users to build a unique model accessible by different users with various points of view, instead of building several sub-models whose management is too hard to complete. Those concepts of view and viewpoint were implemented by VBOOL, the language which propose a new relationship "the visibility". VBOOM, the analyze/design method, integrates those concepts in an object-oriented modeling. The aims of this work are, firstly to propose a new representation of the visibility relationship of VBOOL in UML standard language for modeling and specifying object-oriented systems. Secondly, to complete UML by an oriented viewpoint method to get a complete software engineering process. The definition of this method is based on VBOOM method. This method is called U_VBOOM, which represents an adaptation of VBOOM in UML. The new representation of the visibility relationship encourages the multi-targets code generation and improve the process of development proposed by the VBOOM method.

1 INTRODUCTION

Recently, two aspects have received a lot of attention in object-oriented development: the emergence of Unified Modeling Language (UML) as an unified notation for object-oriented analysis and design, and by integrating viewpoint approaches to software development.

The UML (Rumbaugh, 1999) can be seen as the successor of the wave of object-oriented analysis and design methods that appeared in the late 80s and early 90s. It unifies the methods of Booch (Booch, 1994) , Rumbaugh (OMT) (Rumbaugh, 1989) , and Jacobson (OOSE) (Jacobson, 1992) . The UML is a standard language for modeling and specifying object-oriented systems. It gives notations for describing a system in various views, but does not define any specific process for software development, beyond some preliminary process description reported, for instance, in (Krutchen, 2000) (OMG, 2001) .

The introduction of viewpoint approaches to software development provides several improvements in complex system modeling (Carré, 1991) (Finkelstein, 1993) (Mili, 1999). In fact, it enables the users to build a unique model accessible by different users with various viewpoints, instead of building several sub-system whose management is too hard to complete. The concept of viewpoints was first introduced by Shilling and Sweeny (Shilling, 1989) as a filter of a global interface of the class, but the views are not separable or separately reusable. Harrison and al. proposed subject-oriented programming as a way to build integrated “multiple view” applications by composing application fragments, called subjects, which represent compilable and possibly executable functional slices (Harrison, 1993) .

To this effect, the VBOOL (View Based Object Oriented Language) language has been defined which integrate the new relation "**the visibility**" and its derived mechanisms (language that extend Eiffel) (Marcaillou, 1995) . To implement those concepts in object-oriented methodology, the VBOOM (View Based Object Oriented Method) method has been defined which extends the BON's method (Coulette, 1996) (Kriouile, 1995) . VBOOM propose 3 stages. The first one defines the system dictionary. The second one allows designer to develop their own model separately. This stage meets the need of users to focus on their specification.

The aims of this work is, firstly to propose a new representation of the visibility relationship of VBOOL in UML (Unified Modeling Language)

(Jacobson, 1993) (OMG, 2001) (Rumbaugh, 1999). Secondly, to complete UML by an oriented viewpoint method to get a complete software engineering process. The definition of this method is based on VBOOM method.

This article is organized as follows: in section 2, we define briefly the visibility relationship and it's derived concepts in UML. The section 3 deal with present principles of the VBOOM method under the UML standard (named thereafter U_VBOOM). We conclude by a survey of work done and a presentation of its perspectives after a presentation of the relative works.

In this paper, we are going to illustrate our subjects with the Media library Management example. The specifications of this example are more explained in (Lopez, 1998) . The **Media library** system must allow its members to consult and to borrow various types of support: books, video and audio disks, audio CD, etc. Only one member of the library can borrow books, reviews, etc. The borrow is limited in time. The potential users of the **Media library** system are: the **librarian** who manages the loans, the **person in charge of adhesions** who will add and withdraw members, the **person in charge of exemplaries** who will seize the new exemplaries and to withdraw those damaged, and finally the system engineer who ensures the good exploitation of the system for the users. According to the use types, the Media library system will be considered, as a set of COUNTS ADHERENTS, or of EXEMPLARIES, or a means to facilitate the LOANS. Thus, we identify 4 classes of the system: "Media_Library", "Loans", "Exemplaries" and "Counts_Adherents".

2 THE VISIBILITY RELATIONSHIP: ASSOCIATED MECHANISMS AND SYNTAX

By **visibility**, we mean that an entity can be seen upon several angles. It's the fact of a satellite which can be studied under the mechanic angle, thermal or informative one. The syntax and the semantic of this concept is detailed in (Marcaillou, 1995) , we summarize here the principal characteristic.

A **view** is an abstraction of the model. It constitutes the unity of visibility, it is the result of factorizing user's needs.

A **viewpoint** is the sight that has a user to the model. It 's a combination of views.

A **flexible class** is a class that declares more than two views, its instances which must specify a particular viewpoint take various appearances. In the Figure 1, the "Media_Library" class is a flexible class, which owns 3 views ("Loans", "Exemplaries", "Counts_Adherents").

The graphic representation of the flexible class and the visibility relationship are illustrated respectively by the "flexible" stereotype added to the class symbol of UML and the stereotype "seen_as" added to the inheritance relationship symbol of UML (Figure 1).

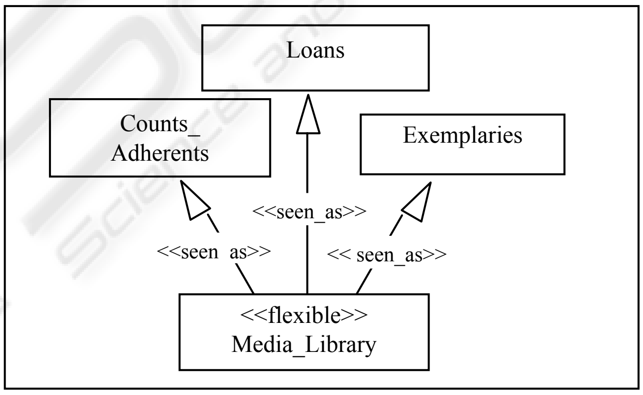

Figure 1: Visibility relationship and flexible class

The syntax proposed by VBOOL to declare views is the following:

```
flexible class Media_Library
  feature...
-- declaration of views
    seen_as Loans
    seen_as Exemplaries
    seen_as Counts_Adherents
.....
end class  -- Media_Library
```

In a flexible class the definition of new features implies specification of their export status. Some features can be exported to some client’s classes, other should be used only by specific instances with particular viewpoint on the flexible class. This restriction of visibility is expressed by the "**view_feature**" keyword.

```
flexible class Media_Library
feature...
-- Views declaration
...
-- View feature declaration
view_feature Loans, Exemplaries, Counts_Adherents
    display do ... end
...
end class -- Media_Library
```

Instances of a flexible class are defined by specifying a viewpoint as follows:

Media_library_librarian: Media_Library (Loans, Exemplaries, Counts_Adherents);

The "Media_library_librarian" is an instance of "Media_Library" having the viewpoint (Loans, Exemplaries, Counts _Adherents). Features declared in "Loans", in "Exemplaries" and in "Counts _Adherents" classes and those specified in "view_feature Loans, Exemplaries, Counts _Adherents" or in "feature" clauses of "Media_Library" class are accessible by "Media_library_librarian" instance .

The object to use the viewpoint is to define access rights to the model. If we consider **librarian**'s access for example the views "Loans", "Exemplaries" and "Counts_Adherents" concern him. Contrary to the **person in charge of members**, the **librarian** must not have the right to change the adherent's feature "block_count_adherent", so to call this feature of the "Counts_Adherents" view. To solve this problem, the "mutual exclusive views" concept has been introduced, that means to specify the views that cannot be simultaneously in the same viewpoint. Thus, in the Media library Management example, we can create two views inheriting from "Counts_Adherents": the views "Mod_Counts_Adherents" and "Not_Mod_ Counts_Adherents". These two views are in mutual exclusive (Figure 2): that is to say "Mod_Counts_Adherents" view that will have access the **person in charge of exemplaries** and "Not_Mod_Counts_Adherents" view that will have access the **librarian.** Of the same way, the **librarian** must not have the right to change the "number_available_exemplary" feature of a "Exemplaries" view, contrary to the **person in charge of exemplaries** who can add new exemplary or to withdraw the damaged exemplaries by invocation the two features "add_exemplary" or "withdraw_exemplary" of the "Exemplaries" view. Thus, we can also create two views inheriting from "Exemplaries", "Mod_Exemplaries" view that will have access the **person in charge of exemplaries** and "Not_Mod_Exemplaries" view that will have access the **librarian** (Figure 2).

VBOOL propose others concepts such as:

- visibility derivation which enables a class to see a flexible class upon a particular viewpoint,
- viewpoint evolution, polymorphism.. etc.

3 STAGES OF U_VBOOM METHOD

VBOOM is a method of analyze/design, which integrates the multiview approach in a coherent and deductive method (Coulette, 1996) (Kriouile, 1995) . VBOOM provides users the possibility to:

- deal strategically with problem space by identifying user's needs,
- improve the interaction expert/user and designer,

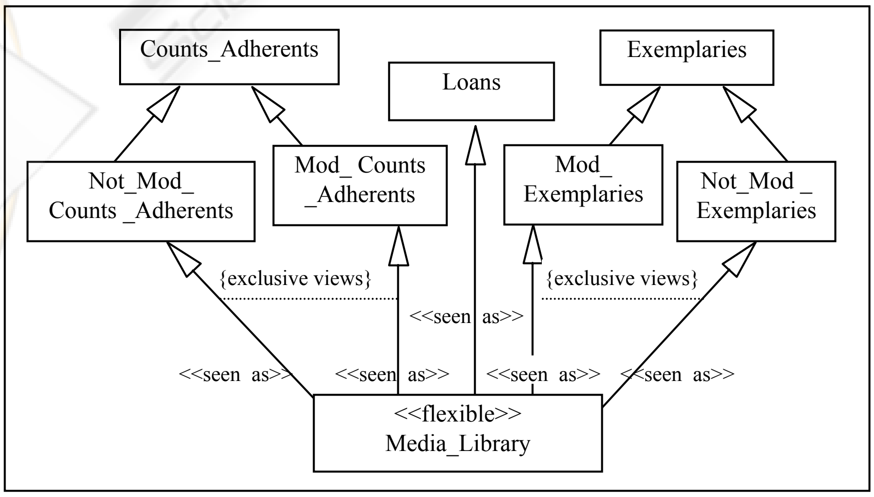

Figure 2: Example of mutual exclusive views

- establish partials models, operating different knowledge of the system. Those models, which make design easier, are called "model's view". Each one concern a specific viewpoint,
- generate a unique model integrating different view models.

Like has been made for the OMT (Rumbaugh, 1989), OOSE (Jacobson, 1992), BOOCH (Booch, 1994) methods, the VBOOM method must take in account the standard UML, i.e. to integrate the concepts and the notations used in the unified modeling language in the VBOOM method, by using its possibilities of specialization and extension (notably the stereotypes) (Hair, 2001) (Hair, 2002). The U_VBOOM method presents an adaptation of VBOOM to UML. The development model of U_VBOOM is iterative, incremental, and piloted by the use case of UML (Jacobson, 1992) (Jacobson, 1993).

As U_VBOOM is an adaptation of VBOOM to UML, U_VBOOM method proposes two kinds of models: static and dynamic one to describe the system and its behaviors. The establishment of those models is an incremental process of 3 stages. The first stage consists of defining model components, the second one enables designer to develop their commonly or not, The final stage has the goal to generate a coherent document (global model, dictionary of components, scenarios).

3.1 Stage1: global analysis

The object of the first stage is to define model components. It is a stage of global specification of development by U_VBOOM. It consists in providing a precise description of the different needs of the system users.

To express the users needs through the system, U_VBOOM proposes the use cases of UML (Figure 3). The views elaboration of the system is supported by the use cases description technique in actions (Hair, 2001) (Hair, 2002). This technique consists to describe the use case as an action sequence that will make an actor to achieve his goal. An action is an effect produced by an actor acting in way given on the system or by the system itself (Dano, 1997). Every action has a number and a label who are indicated in the “Decomposition in actions” column (Table 1).

The use cases identified for the actors and the intersection of their actions are going to permit to cut the viewpoints in views. A view corresponds to actions list to a given viewpoint or result of actions intersection of viewpoints group (Figure 4).

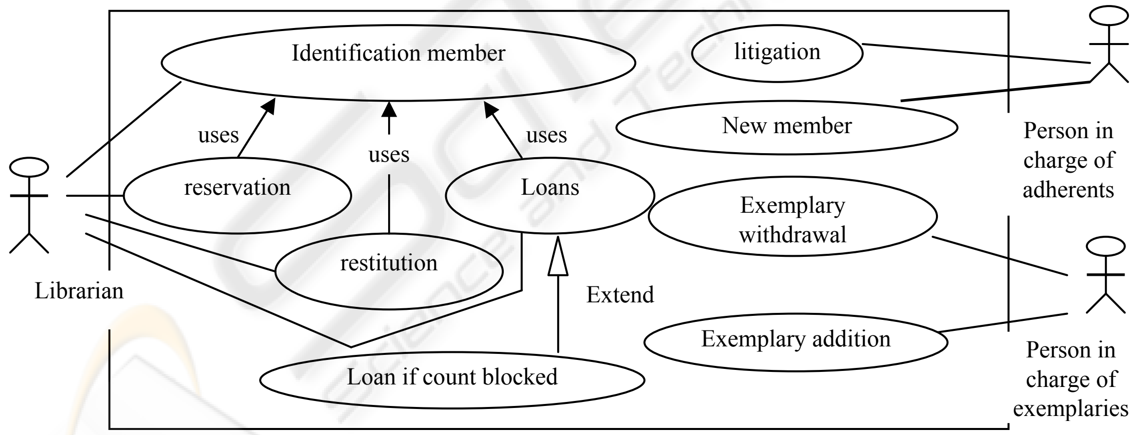

Figure 3: Use case diagram of MEDIA LIBRARY system

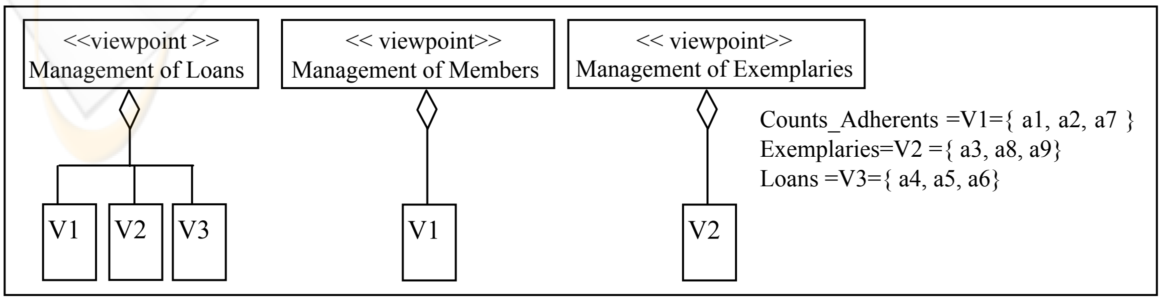
<<viewpoint >> Management of Loans
<< viewpoint>> Management of Members
<< viewpoint>> Management of Exemplaries
Counts_Adherents =V1={ a1, a2, a7 }
Exemplaries=V2 ={ a3, a8, a9}
Loans =V3={ a4, a5, a6}
V1
V2
V3
V1
V2

Figure 4: Viewpoint diagrams of the MEDIA LIBRARY system

Table 1: Decomposition of the different use cases in actions

Actors	Use case	Decomposition in actions
Librarian	Loan	**a1**: To identify an adherent **a2**: Count adherent (To check right loan for member) **a3**: To seek for an exemplary **a4**: To treat an exemplary (to validate the output of an exemplary) **a5**: To treat adherent (to indicate the loan by adherent)
	Reservation	**a1**: To identify an adherent **a2:** Count adherent (To check right loan for member) **a3**: To seek for an exemplary **a6**: To reserve an exemplary
	Restitution	**a1**: To identify adherent **a3**: To seek for an exemplary **a4**: To treat an exemplary (to validate the input of exemplary) **a5**: To treat adherent (to indicate the return of an exemplary)
	Loan if counts blocked	**a1**: To identify adherent **a2**: Count adherent (To check right loan for member)
	Identification member	**a1**: To identify adherent
Person in charge of adherents	New adherent	**a 2**: Adherent account (to Add adherent)
	Litigation	**a1**: To identify adherent **a2**: Count adherent (Blocked count adherent) **a7**: To inform adherent
Person in charge of exemplaries	Exemplary addition	**a3**: To seek for an exemplary **a8**: To add copy
	Exemplary withdrawal	**a3**: To seek exemplary **a9:** To withdraw the damage exemplary

The global analysis stage continues to identify objects and classes belong to the with problem domain. The classes are discovered, scenarios after scenarios, by means of objects who, in collaborating, achieve use cases. It leads to the development of the class diagrams (Figure 5) and the object diagrams.

Finally, the packages can be identified to organize the modeling elements. The identified packages are gotten by according to the logical criteria use type of an actor. These packages are going to become thereafter the sub-system (named model's view) during the second stage of the U_VBOOM method (Figure 6).

The classes constituting a package represent a part of class diagram of the system. This part is defined in respecting the following rules:

- the only views (classes) of the package are those of its associated viewpoint,
- the classes joined by the customer relationship to one of the package classes,
- the classes jointed by inheritance of each package classes.

3.2 Stage2: The sub-systems design

The global analysis of the U_VBOOM method is elaborated and it will be enriched in second stage. The design of sub-systems (packages identified in the global analysis stage, named model's view) permits the translation of the analysis model. These sub-systems constitute an essential artifact of the second stage of the U_VBOOM method. Indeed, the cutting out the solution space of the problem in sub-systems permits to land the global system design to sub-systems designs. The design of sub-systems can be made in a disparate and autonomous way and be led in parallel or sequential.

The second stage of U_VBOOM has for object to achieve the partials class diagrams and to define the partials classes interfaces of every sub-systems (the class interface contains the list of its features). The designer must come back toward to realize the use cases. The scenarios, the collaborations between analysis objects and the class diagrams are refined to get the design classes constituting the partial dictionary of every sub-system. The Figure 7 represents the class diagram (partial) of the **Management of Loans** sub-system.

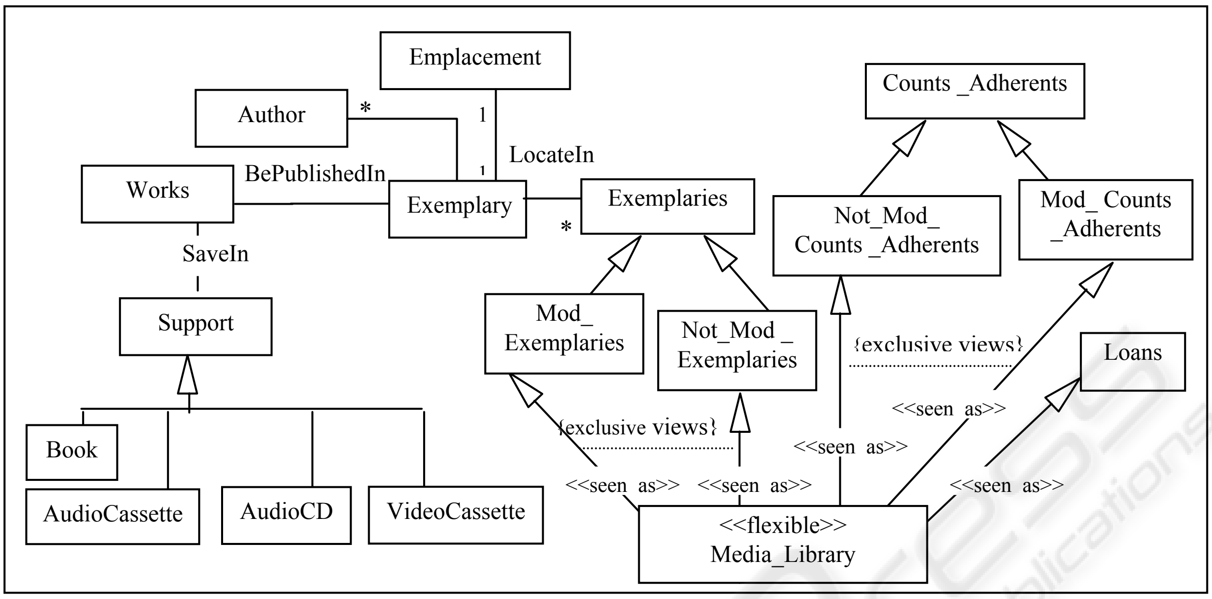

Figure 5: Initial class diagram of the MEDIA LIBRARY system

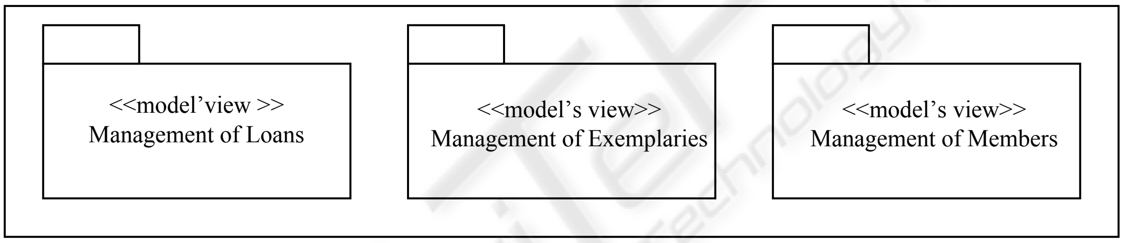
<<model'view >> Management of Loans
<<model's view>> Management of Exemplaries
<<model's view>> Management of Members

Figure 6: The 3 packages of the MEDIA LIBRARY system

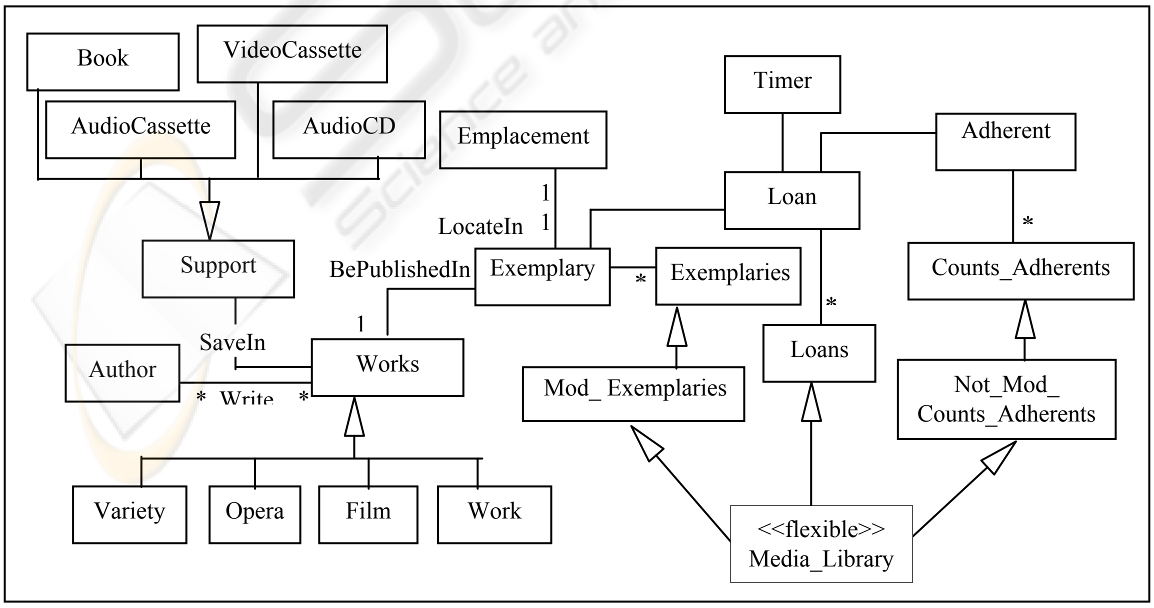

Figure 7: Class diagram of the Management of Loans sub-system

3.3 Stage3: Global model design

The third stage of the U_VBOOM method provides to the process designers to melt the differents partials class diagrams and the differents partials class interfaces of the sub-systems obtained from the second stage of the method. The melting process proposes to the designers the heuristic to manage the conflicts appeared during of this stage (polysemies, synonymies, homonymies...) (Figure 8). In our example, the global model is relatively close to the **Management of Loans** sub-system because this last is predominant in the system, but this situation is not evidently a generality. The sub-systems obtained are tested and validated in order to be coded in the implementation activity.

4 CONCLUSION

The representation of relationship visibility of VBOOM in UML encouraged the integration of UML in VBOOM. The U_VBOOM method that represents the result of VBOOM adaptation under the UML standard is incremental, iterative and piloted by the use cases.

The decomposition of the system in packages, which became the sub-system, by the logical cutting based on use type permitted to land the global system design to the sub-systems designs.

The work describes herein is part of a project which define a methodology of development of components multiview objects. Among the tasks remaining to achieve in this project, we can mention:

- the definition of the composing multi-views notion as regrouping of multiview classes,
- the development of a basis of design pattern supporting the viewpoints approach,
- the realization of an environment support of U_VBOOM.

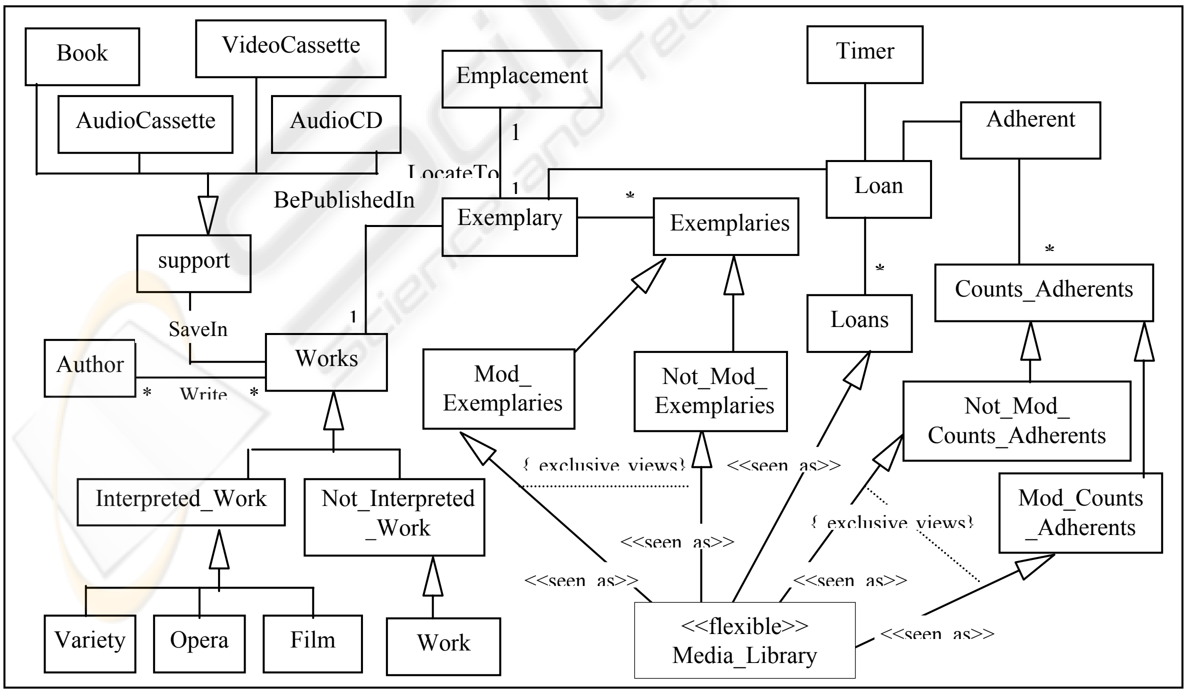

Figure 8: Class diagram of the MEDIA LIBRARY system

REFERENCES

Booch, G., 1994. Object-Oriented Analysis and Design with Applications. Benjamin/Cummings, Redwood City.

Carré, B. 1991. The Point of View notion for Multiple Inheritance. *In ECOOP/OOPSLA'91*.

[Coulette, B., 1996. L'approche par points de vue dans le développement orientée objet de systèmes complexes. *Revue l'Objet, Vol. 10, No. 5, pp. 13-20.*

Dano, B., 1997. An Approach Based on the Concept of Use Cases to Produce Dynamic Object Oriented Specifications. *In the Third IEEE International Symposium on Requirements Engineering.*

Finkelstein, A., 1993. Inconsistency Handling in Multi-Perspective Specifications. *In ESEC'93, Garmish-Paternkirchen , pp. 84-99.*

Hair, A., 2001. VUML : Une méthode d'analyse et de conception orientée objet, intégrant UML et le concept de point de vue. *In ICSSEA'2001, the International Conference on Systems, Software Engineering and their applications.*

Hair, A., 2002. *Un processus d'analyse et de conception unifié basé sur le concept de point de vue. In CARI'02, 6th Africain Conference on Research in Computer Science, pp. 229-237.*

Jacobson, I., 1992. Object-Oriented Software Engineering, A Use Case Driven Approach. Addison-Wesley, Inc..

Jacobson, I., 1993. The Unified Software Development Process. Addison-Wesley, Inc..

Harrison, W., 1993. Subject-oriented programming: a critique of pure objects. *In OOPSLA'93, Washington D.C., pp. 411-428.*

Kriouile, A., 1995. VBOOM, une méthode d'analyse et de conception par objet fondée sur les points de vue. *Ph.D. thesis of sciences faculty, Rabat.*

Krutchen, P., 2000. The Rational Unified Process - An Introduction. Addison-Wesley, Inc..

Lopez, N., 1998. Intégrer UML dans vos projets. Edition Eyrolles.

Marcaillou, S., 1995. Intégration de la notion de points de vue dans la modélisation par objets ; Le langage VBOOL. *Ph. D. Thesis of Paul Sabatier university, Toulouse.*

Mili, H., 1999. View programming of OO applications. *In TOOLS'99, 1999.*

Meyer, B., 1995. Object success - A managers's guide. Prentice Hall - The Object-Oriented Series.

OMG, 2001. Unified Modeling Language (UML), version 1.4, Document formal/2001-09-07, http://www.omg.org/cgi-bin/doc?formal/01-09-67,

Rumbaugh, J., 1989. The Unified Modeling Language Reference Manual. Addison-Wesley.

Rumbaugh, J., 1989. OMT : Modélisation et conception orientées objet. Prentice Hall.

Shilling, J., 1989. *Three Steps to Views. In OOPSLA'89, New Orleans, LA, pp. 353-361.*