A General Framework for Describing the Requirements Engineering Process

C. Rolland, G. Grosz
C.R.I. University of Paris 1 - Sorbonne
17 rue de Tolbiac
75013 PARIS FRANCE
email : {rolland, grosz}@masi.ibp.fr

Abstract :

This article concerns automated guidance of the requirements engineering process in a CASE environment. We see guidance as part of a broader framework for process tracing, method engineering and process control which is sketched in the paper. This framework supports the claim that a process knowledge base cannot be built once and for all but must be constantly improved. This leads to the idea of a process knowledge base composed of a set of process chunks which are progressively defined. This is achieved by using the concepts of a guidance meta-model which is the kernel of the paper.

1. INTRODUCTION

The Requirements Engineering (RE) activity leads to define high level specifications of Information Systems (IS). It is widely acknowledged that the RE process lacks in-depth studies. However, it is also widely accepted that the quality of IS specifications depends on the process by which these specifications are built [1]. Therefore, there is a strong need for guidance of the RE process. This need is not fulfilled by the current CASE tool technology. The main reasons are the following :

- there is no representation of ways of working in the CASE repository. CASE tools are capable of storing, retrieving and manipulating the specifications (graphical or textual) but provide very few hints on how to build the specifications.

- the absence of process representation is explained by limitations of way of working in current methodologies. As far as methodologies are concerned, ways of working are usually described in fuzzy terms, at a very global level. They consist of a list of very coarse grain tasks. For instance, the Entity-Relationship method [15] is most often described as a loop involving the following tasks : definition of entity type, completion of entity type, definition of relationship type and possible completion of relationship type. Such a description is not accurate enough to provide efficient guidance.

- there is no capitalization of experience. CASE tools are not able to adapt according to the way they are actually used by application engineers. Therefore, there is an important loss of expertise.

Based on these limitations, we advocate the following :

- the CASE tool's repository must be extended with a process knowledge base,

- there is a strong need for a more precise, complete definition of ways of working proposed by methodologies, a way of working must be defined at both a coarse grain level and at a very fine grain level.

- in order to allow capitalization of experience, tracing facility is mandatory in a CASE environment [2].

In the context of the ESPRIT project NATURE[1], we propose a process repository as being the core element of a CASE environment as shown in figure 1.

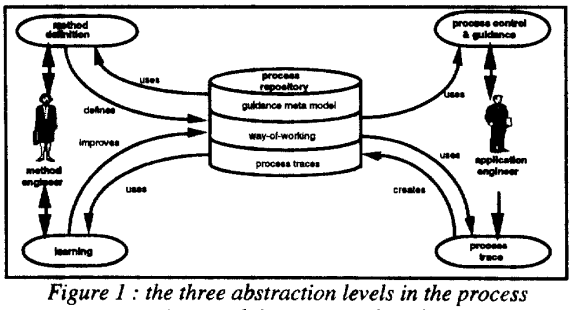

Figure 1 : the three abstraction levels in the process repository and the associated modules

The process repository introduces three levels of abstractions :

- at the lowest level, process traces are recorded. A process trace describes the complete evolution of an IS specification during the development process. There is one process trace for each application developed. A process trace is an instance of a way of working followed by the application engineer.

- at the second level, ways of working are defined. A way of working provides control and guidance to the application engineer to construct IS specifications. It describes, at the type level, what can be done to proceed in the development

1 This work is partly funded by the Basic Research Action NATURE (ESPRIT N° 5363). NATURE stands for Novel Approaches to Theories Underlying Requirements Engineering

0-7803-2129-4/94 $3.00 © 1994 IEEE

process, when, why and also how to do it. A trace is an instance of a way of working.

- at the highest level, the guidance meta-model is defined. It is a general framework providing a set of high level concepts for describing ways of working. A way of working is an instance of the guidance meta-model.

All four modules related to the process repository are sketched in [3] and [4].

The architecture of the repository presented in figure 1 can be related to the plan theory proposed by Wilensky [5]. He proposes to define meta-models as a means to generate a plan (also called model) which can be executed and traced.

There are also some similarities between our approach and the one advocated in Information Resource Dictionary Framework Standard (IRDS) [6].This framework proposes to organize a repository along the classification dimension of semantic data model abstractions. The repository is layered in four levels where level *n+1* constitutes a type level for level *n*. In other words, level *n* is an instance of level *n+1*. Within the IRDS framework, the four levels are : *Application level, IRD level, IRD definition level* and the *IRD definition schema level*. Our approach deals with the various levels of description related to the process of building specifications, whereas the IRDS approach deals with the levels of description related to the specification themselves.

Assuming that the existence of a meta model is related to method definition, ways of working associated to methodology can be precisely described. According to our view, guidance is based on the use of such ways of working and process traces. A way of working provides a means to propose to the application engineer what he/she can do whereas traces are used to learn from the actual use of the way of working. The RE process requires constant improvement, capitalization of experiences. There is a strong need for modularization of the way of working definition. We look at a way of working as composed of process chunks which can be added, removed, modified and improved at any time. This introduces the need for a modular process knowledge base.

The paper is structured as follows. In section 2, we concentrate on the structure of the process knowledge base and present its core element, the guidance meta-model. In section 3, we instanciate the meta-model to the OMT methodology [7]. Conclusions are drawn in section 4.

2. THE GUIDANCE META-MODEL

We start this section by introducing the main characteristics of the guidance meta-model we propose. We then provide a global overview of the meta-model and end with a detailed presentation of each concept.

2.1. MAIN CHARACTERISTICS

In the literature, several classes of process models have been proposed. Dowson [8] identifies three major classes: activity-oriented models, product-oriented models and decision-oriented models. We strongly believe that decision-oriented models are the most appropriate for describing the RE process. According to this class of models, the successive transformations of the specifications are looked upon as consequences of decisions. The process model of the DAIDA project [9] and of [10] fall into this category. Such models are semantically more powerful than activity-oriented or product-oriented models because they explain not only how the process proceeds but also why transformations happen.

The guidance meta-model introduced in this paper belongs to this last category. However, even though existing decision-oriented models offer limited guidance to the designer, very little is said about when and how to decide on what. Our guidance meta-model was first sketched in [11] and refined in [12]. A first implementation has been done in the ALECSI prototype [13].

The particular RE process modeling approach we chose emphasises the notion of decision within the context it is made. Application engineers react *contextually* according to the domain knowledge they acquire and react by analogy with previous situations they have been involved in. In order to take into account the very nature of RE we have chosen to emphasise the contextual aspect of decisions. This approach strongly couples the context of a decision to the decision itself, otherwise we lose information about the decision. Our approach aims at capturing not only how activities are performed during the RE process but also why these activities are performed (the decisions) and when (the decision contexts).

2.2. THE GUIDANCE META-MODEL : AN OVERVIEW

Figure 2 presents a global view of the guidance meta-model using some ER binary like notations.

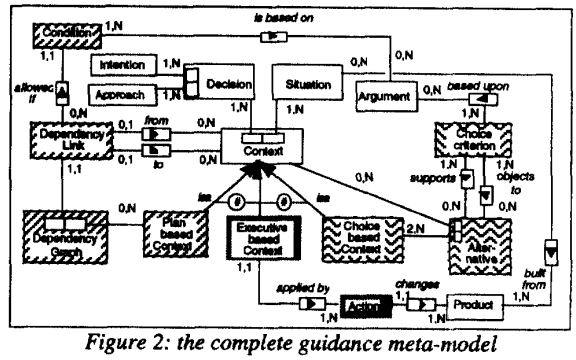

Figure 2: the complete guidance meta-model

The central concept of this meta-model is the one of *context*. A *context* allows to describe a step of the RE

process : in a situation the designer has to make a decision in order to progress in the RE process, therefore, a context is the coupling of a situation and a decision. This association is very meaningful. A decision is not sufficient in itself, it needs to be associated to the situation where it applies. For instance, it does not make sense to specialize an entity, if this entity does not exist. Similarly, one cannot perform integration if there is only one module in the specification.

We recognize different types of decisions.

First, the ultimate purpose of a decision is to act, to modify the current specification. For instance, having the intention to add an entity type "customer" in a ER specification leads to perform an action creating the entity type. This type of decision and its associated situation is represented by what is called *executive based context.*

Second, some decisions can be fulfilled in several different ways. For instance, if one wants to define the key of an entity type, he/she can either define a simple key or a composite one. This calls for what we refer to as *choice based context.*

Third, some decisions require a number of other decision to be made before being completely fulfilled. For instance, when building an ER specification, one has to define entities and relationship between entities. In the meta-model, this is represented by what we called a *plan based context.*

2.3. THE GUIDANCE META-MODEL : PRESENTATION

In this section, the different concepts of the guidance meta-model are described in turn.

2.3.1 The concept of context

As already mentioned, the central concept of this meta-model is the one of *context*. A *context* is build from a *decision* and a *situation.*

A *situation* is most often a part of the specification it makes sense to make a decision on. *Situations* are built from parts of the specification undergoing the RE process. At the beginning of the process, the specification is made up of the collection of the users' requirements. At the end, the output is a specification of the IS that satisfies some quality criteria. *Situations* in RE can be of various granularity levels. They can be either atomic like an attribute in a class ; Or they can be coarse-grained like the whole specification under development. Moreover, *situations* can also be built from existing, reusable parts of previously developed specifications. [14] presents a model of the IS specification and details the building of situations from its parts.

A *decision* reflects a choice that an application engineer can make at a given point in time of the development process. A *decision* encapsulates two aspects, namely *intention* and *approach.*

An *intention* expresses what the engineer wants to achieve, it is a goal. An *intention* can be global or local allowing consequently various levels of granularity in the decision making process. For instance, a global *intention* can be "define an OMT schema" whereas a local *intention* can be "add new attribute".

An *approach* characterises ways to fulfil an intention. Considering IS development, we currently recognize two different *approaches*, namely bottom-up and top-down. Following a bottom-up *approach* means to define first the component and then the composite concepts, whereas following a top-down *approach* means to first define composite elements and then their components.

A *decision* is defined as an *intention* - to achieve - possibly following an *approach*. For example, in the OMT methodology, the *intention* of "defining a class" will be implemented differently depending on the *approach* : top down approach or bottom up. In the former case, the class can be defined from scratch whereas in the latter case, the class is defined by aggregating its elements - attributes, operations, etc.. Therefore, the same *intention* can be related to several *approaches*. However some *intentions* only exist within a single *approach*. For instance, the *intention* "aggregate elements into class" makes sense only in the bottom up *approach.*

A *context* is the association of a *situation* and a *decision* which can be made on this very situation. It should be clear now that a *situation* can be associated to several *decisions* - e.g. the situation "class" is associated to the decisions "define" and "refine" in two different contexts. Similarly, a *decision* can be made on several *situations* - the decision "refine" can be made on a *situation* made up of a class as well as an attribute.

2.3.2 The different type of contexts

a) Executive based context

At the most detailed level, the execution of the RE process can be seen as a set of transformations performed on the specification under development, each transformation being the application of a deterministic action - e.g. create a class, an attribute, delete a class, etc.. This leads to introduce the concept of *executive based context* as a specialization of the concept *context*. Such a context implements a decision, it allows to represent the change of the specification under development. Therefore, an executive *based context* is associated to an *action*. An *action* performs a transformation on the specification under development. It is the materialization, the implementation of a decision. Performing an action changes the specification and may generate new situations, which are themselves subjects of new decisions. Figure 3 presents two contexts related to the ER methodology and shows how a new situation arises - and how a context can be applied - after the execution of an executive based context

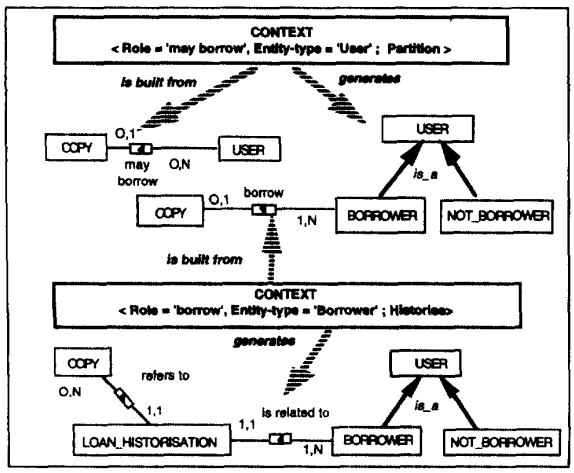

Figure 3 : examples of executive based contexts

b) Choice based context

When building an IS specification, an application engineer may have several alternative ways to fulfil a decision. Therefore, he/she has to decide the most appropriate one, to refine the initial decision according to a set of possible choices. In order to describe such a piece of knowledge, we introduce a second specialisation on the concept *context*, namely : *choice based context.*

A *executive based context* leads directly to modify the specification whereas a *choice based context* does not. In the latter case, a set of alternatives have to be studied and one has to be selected. To help the selection process, *arguments* and *choice criteria* are introduced.

In the guidance meta-model, the various alternatives of a *choice based context* are represented in the *alternative* relationship. Alternatives of a *choice based context* are contexts too, thus contexts may share an *alternative* relationship, leading to alternative - based hierarchies of contexts. Therefore, a *choice based context* can be used at both a very high level of detail - the selection of an approach can be expressed with such a context - and a low level of detail - the selection of the type of an attribute would also be expressed with such a context.

An *argument* allows to motivate the making of a decision in a given situation or to object to this decision.

Choice criteria are associated with alternatives in order to help the application engineer in the decision-making process by supporting or objecting to the different *alternatives* of a given *alternative based context*. These *choice criteria* are combinations of the arguments associated with the alternative. A choice criterion provides priority rules to select one *alternative* among several depending on the *arguments*.

For example, OMT advocates that the application engineer has to validate the classes he/she created. When doing so, he/she can either validate the need for this class, delete it, or retype the class into an attribute, an association or an operation. Thus, the context <class, validate class> is a *choice based context.* In the following, we exemplify some arguments. Associated with the first alternative - <class, validate the need> we have : (supporting) Arg1 : "this class explicitly refers to a real world phenomenon relevant to the IS", (objecting) Arg2 : "this class does not have any specific behavior nor any attributes". Associated to the second alternative - <class, delete class> we have : (objecting) Arg1. Concerning the third alternative, the supporting arguments are : (supporting) Arg3 : "this class has no specific behavior and represent an aspect of an existing class". OMT emphazises readability of the specifications, therefore a *choice criterion* would suggest first the contexts leading to delete or to retype an existing class. This context is graphically represented as follows:

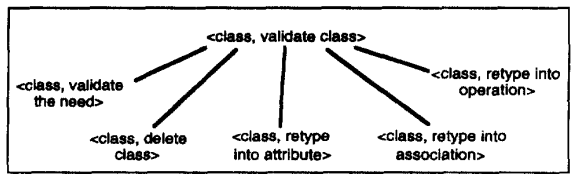

Figure 4: an example of a choice based context

c) Plan based context

In order to fulfil an intention, an application engineer may be required to make a set of decisions, he/she has to follow a plan. To this end, we introduce a third specialization of context, namely, *plan based context.* A plan based context allows to decompose a decision into a set of decisions.

As the loop on figure 2 indicates, a *plan based context* is composed of several contexts - being them executive, choice or plan based context. This association expresses the fact that in order to fulfil an intention a set of contexts has to be applied. The ordering of the contexts is defined by a graph - named a *dependency graph.* There is one graph per *plan based context.* The nodes of this graph are contexts. A link - called a *dependency link* - defines in both a dependency between two contexts and a possible transition between the two contexts. Based on some arguments, a condition can be associated to a link. It defines when the transition can be performed. Note that a graph can be followed in different ways. Combined with approaches, this allows more flexibility in guidance than traditional process models.

Similarly to *choice based context, plan based context* can be use to describe the RE process at a very high level of detail - e.g. how to build an OMT specification - as well as at a very low level of detail - e.g. how to build a class.

For instance, in OMT, the analysis process is decomposed into object modeling, dynamic modeling and functional modeling. Within our guidance meta-model, this is represented as follows. The context <problem statement, perform analysis> is a *plan based context* composed of the following list of contexts : <PS, build Object Model>, <PS

+ OM, build Dynamic Model> and <PS + OM + DM, build functional model>. This is represented as follows.

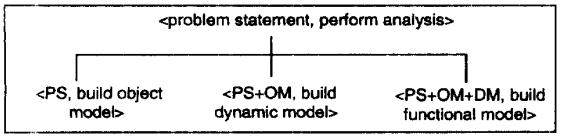

Figure 5 : an example of a plan based context
PS stands for problem statement, OM, DM, FM stand for object, dynamic and functional models respectively.

Concerning the graph associated to the context above, the OMT methodology advocates to start with object modeling, continue with the dynamic modeling - if at least two classes have been defined - and then either complete object modeling or perform functional modeling. After having finished the construction of the functional model, the application engineer can either stop or add elements in the object model if he/she follows an iterative approach or if a problem statement has been modified or a new operation has been introduced in the functional model. The dependency graph associated to the context <problem statement, build object model> is depicted in figure 6.

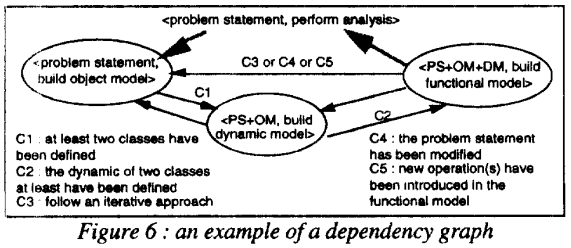

Figure 6 : an example of a dependency graph

3. INSTANCIATING THE META-MODEL TO THE OMT METHODOLOGY

The following section is based on readings of the OMT book [7]. The aim is to exemplify the instanciation of the guidance meta-model introduced in the previous section to OMT. We first describe the OMT methodology at a very high level of granularity and successively refine some elements until we reach the elementary process part. This methodology does not advocate an alternative approach, it is mostly top-down.

At the top level, OMT advocates to iterate between analysis and design and then perform implementation. Within our framework, this is expressed by a plan based context. This context is composed of three contexts having as intentions to perform the activity mentioned above. This is represented as follows.

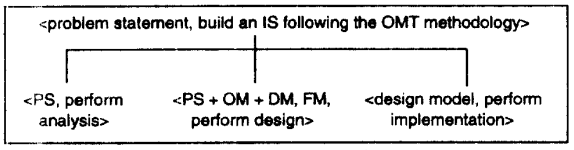

Figure 7: the top level context describing OMT

The associated graph is the following :

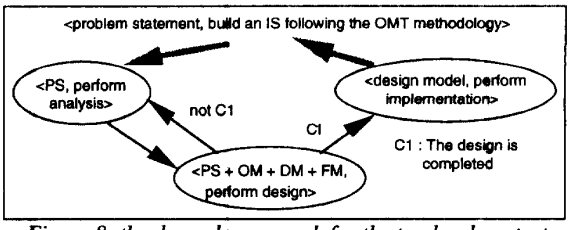

Figure 8: the dependency graph for the top level context

Each component context should be described in turn. For the sake of brevity, we concentrate on the context <PS, perform analysis>. This plan based context has been presented in the previous paragraph, We just recall its structure.

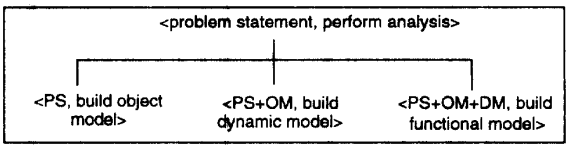

Figure 9: the context dealing with analysis

The context <problem statement, build object model> is a plan based context as well. It is composed of the following contexts : <PS, define class>, <class, define attribute>, <class + attribute, build the data dictionary>, <set of classes, define association>, etc. As for any plan based context, a dependency graph is defined. In this case, the graph comprises description of the sequencing between class definition, attribute definition data dictionary construction and association definition and loops on some composite contexts because they can be applied several times.

Let us now detail the context <set of classes, define association>. When defining an association, one has to use first <set of classes, identify association > and then <set of class + association, validate association>. Therefore <set of classes, define association> is a plan based context.

Let us now concentrate on the context <set of classes, identify association>. In OMT, one can either reuse an existing association and extend its relationship to new classes or create the association. In the latter case, several alternatives must be considered. The association can either be one-to-one, multiple, etc.. All this knowledge about the identification of an association is represented with the following hierarchies of choice based context.

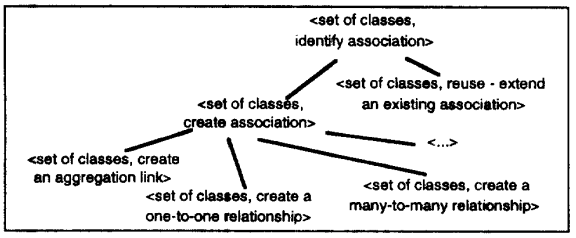

Figure 10: hierarchy of contexts for association identification

4. CONCLUSION

In this paper, we present a guidance meta-model that aims at offering a framework to precisely describe the RE process at various levels of details. The building block of this meta-model is the concept of context. A context is composed of a situation and a decision. Within a situation, the application engineer is able to make a decision. In meta-model, the concept of context is refined in three ways. To summarize, executive based contexts describe how to implement a decision, they are associated to some action modifying the specification under development. Choice based contexts allow to represent the various alternatives, they refine decisions. Arguments and choice criteria help the selection of the appropriated alternative. Plan based contexts provide a means to decompose a context into several other contexts and to express the chaining among the composite contexts. Both plan and choice based contexts can both be used at a very high and a very low level of granularity.

We claim that any way of working related to any methodology dedicated to RE can be described with a hierarchy of contexts. A hierarchy is a set of contexts related together by either a composition link (through plan based context) or an alternative link (through choice based context), the leaves of the hierarchy being executive based contexts. Such a hierarchy and the associated graphs allow the definition of ways of working that embed IS engineering heuristics. To be able to express heuristics is an important improvement in way-of-working definitions since it allows to capitalize experiences and permit that application engineers share their practical knowledge and methods.

In the paper, we took examples based on the ER and OMT methodologies. Other methodologies have been described within our framework (e.g. OOA [16], SASD [17], OOD [18], O* [19], etc.). We are currently working on the implementation of the meta-model based on the framework sketched in section 1. The first results clearly show the effectiveness of our approach : the application engineer is truly guided. However, some problems remain while trying to mix various approaches in the same session, i.e. how to switch from a top-down approach to a bottom-up approach. Another important direction for future research comprises the study of a methodology to help the instanciation of the guidance meta-model.

5. REFERENCES

[1] : Humphrey W.S : "Managing the Software Process", Addison Wesley, Reading MA, 1989.

[2] : Ramesh B., Edwards M. : "Issues in the Development of a Requirements Engineering", Proc. IEEE Symp. on Requirements Engineering, San Diego, Ca, 1993.

[3] : Jarke M., Pohl K., Rolland C., Schmitt J.R.; "Experience -Based Method Evaluation and Improvement: A Process Modelling Approach", submitted paper.

[4] : G. Grosz : "MENTOR : a Step Forward in Guidance for Information System Development", Proc. NGCT94, Uttrecht, The Netherlands, 1994.

[5] : Wilenski J., "Planning and Understanding", Addison Weslay, 1983.

[6] : ISO-IEC 10027 : "Information Technology - Information Resource Dictionnary System (IRDS) - Framework", ISO/IEC International Standard, 1990.

[7] : J. Rumbaugh, M. Blaha, W. Premerlani, F. Eddy, W. Loresen : "Object-oriented modeling and design", Prentice Hall international, 1991.

[8] : Dowson M. : "Iteration in the Software Process"; Proc 9th Int Conf on "Software Engineering", Monterey, CA,1988.

[9] : Jarke M., Mylopoulos J., Schmidt J.W., Vassiliou Y.; "DAIDA - An Environment for Evolving Information Systems"; ACM Trans. on Information Systems, Vol. 10, No. 1, 1992.

[10] : Potts C.: "A Generic Model for Representing Design Methods"; Proc 11th Int. Conf. on Software Engineering, 1989.

[11] : Grosz G., "Formalisation des connaissances réutilisables pour la conception des systèmes d'informations", Thèse de doctorat de l'université Paris 6, Dec. 1991

[12] : Rolland C.: "Modeling the RE Process", Proc. Fino-Japanese Seminar on "Conceptual Modeling", 1993.

[13] : Rolland C., Cauvet C. : "ALECSI : An Expert System for Requirements Engineering", in "Advanced IS Engineering", R. Andersen, J Bubenko, A. Solvberg (Eds), Springer Verlag, 1991.

[14] : Schmitt J.R.: "Product Modeling in Requirements Engineering Process Modeling", IFIP TC8 Int. Conf. on "Information System Development Process", Prakash N., Rolland C., Pernici B. (eds.), North Holland (pub.), 1993.

[15] : Chen P.P. : "The Entity-Relationship Model : Toward a Unified View of Data"; ACM Trans. on Database Systems, 1(1):9-36, March 1976.

[16] : Coad P., Yourdon E : "Object Oriented Analysis", 2nd ed., Yourdon Press, Englewood Cliffs, N.J., 1991.

[17] : Yourdon E. : "Modern Structured Analysis", Prentice-Hall, Englewood Cliffs, N.J., 1989.

[18] : Booch, G. : "Object-Oriented Design with Applications", The Benjamin/Cummings Pub. Comp. Inc., Redwood City, California, 1991.

[19] : Brunet J. : "Modelling the World with Semantic Objects", IFIP Conf. on "The Object-Oriented Approach in Information Systems", Rolland C., Moulin B., VanAssche F. (eds), North Holland(pub), 1991.