ALECSI : AN EXPERT SYSTEM FOR REQUIREMENTS ENGINEERING

C. CAUVET(1) C. PROIX(2) C. ROLLAND(3)

(1) Université PARIS 1 IAE, 162 Rue St Charles
75015 PARIS FRANCE
(2) CRIL, 146 Boulevard de Valmy
92700 COLOMBES FRANCE
(3) Université PARIS 1, 17 Rue de la Sorbonne
75231 PARIS FRANCE

ABSTRACT

The paper aims at presenting a conceptual modelling expert system so-called ALECSI to support requirements engineering. The tool takes user requirements expressed either with natural language sentences or in graphical form as input and produces conceptual structures. During this process, ALECSI supports both knowledge engineering tasks (acquisition, modelling, validating...) and process engineering tasks (guidance, explanation...). The last section of the paper simulates an ALECSI session in order to illustrate with examples the guidance provided by the tool during the requirements engineering phase.

1. INTRODUCTION

The design of large Information Systems (IS) is becoming increasingly more difficult as user requirements become broader and more sophisticated. The major response to the complexity that is exhibited by more contemporary data-intensive applications and the subsequent problem of designing an automated IS, is the emergence of more and more advanced system design methods.

Each method consists in **models** for system specification, a **process** by which specifications are constructed and a set of CASE **tools** to aid design.

The need for modelling techniques by which IS may be described in high level terms in the earlier phases of IS development has been recognized, in industry, business and administration. This has caused the introduction of various conceptual models that have proved to be extremely useful. A number of semantic

models such as SHM [SMIT.77], SHM+ [BROD.82], TAXIS [MYLO.80], REMORA [ROLL.79], SDM [HAMM.81], CIM [GUST.82] have been proposed to capture the real-world semantics with more preciseness and naturalness.

A conceptual model allows to build-in the IS specification e.g. the conceptual schema in high level terms. The route to reach the conceptual schema e.g. the conceptual modelling process has the purpose of abstracting and conceptualizing relevant parts of the application domain. This is guided by requirements.

The term **Requirement Engineering** [DUBO.89], [HAGE.88] will be used in this article to refer to this part of the IS development that invokes investigating the problems and requirements of the users community and developing a specification of the future system. The succeeding phase where the specification serves to realize a computerized system may be called **System Engineering** (Figure 1).

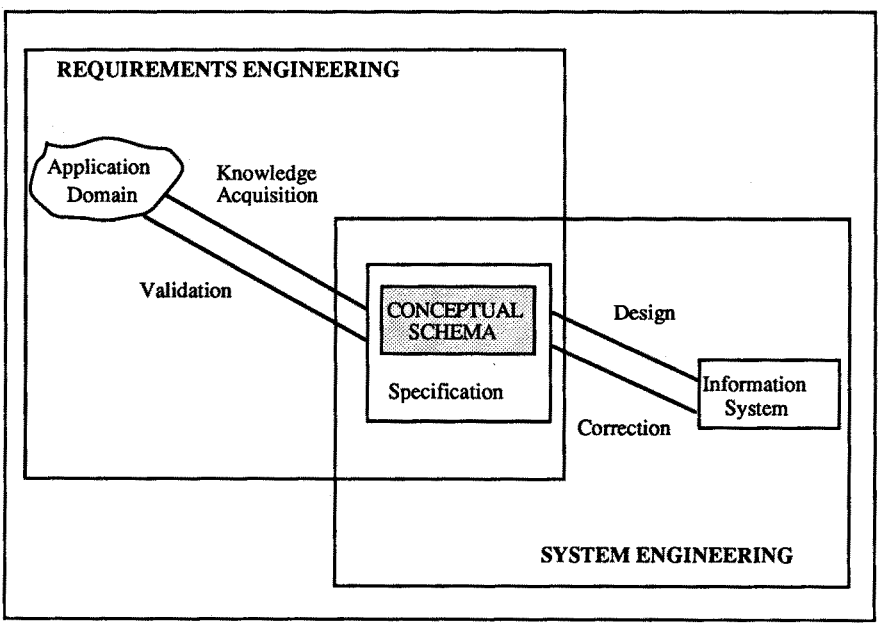


Figure 1 : Requirements engineering and system engineering

The paper aims at presenting a conceptual modelling expert system so-called Alecsi to support requirements engineering.

The key point of Alecsi is its emphasis on process support. Alecsi is providing guidance and help to the designer when he travels the route to reach the conceptual schema. It takes user requirements expressed either with natural language sentences or in a graphical form as input and produces conceptual structures consistent with given sentences or graphs.

From that point of view Alecsi differs from a number of commercialized CASE tools (IEW, EXCELERATOR, CASE ORACLE...) which assist the designer in the management of the specification (the conceptual schema description) but do not aid in the process of constructing the specification. Such tools provide facilities to enter elements of the specification through their diagrammatic expression in a data dictionary, to check their consistency and completness and to document them.

Conversely Alecsi focuses on supporting the designer in the conceptual modelling process i.e. the requirements engineering process of constructing the specification by abstracting and conceptualizing relevant parts of the application domain.

Section 2 presents what is meant by Requirements Engineering and the different ways Alecsi intents to support it in an automated manner. This leads to the description of the functional architecture of Alecsi.

Section 3 describes the three levels of knowledge representation supported by Alecsi.

Section 4 simulates an Alecsi session in order to illustrate with examples the guidance provided by the tool during the requirements engineering phase.

2. REQUIREMENTS ENGINEERING SUPPORT IN ALECSI

2.1 KNOWLEDGE ENGINEERING AND PROCESS ENGINEERING

As shown in figure 1, Requirements Engineering consists of knowledge acquisition and the validation cycle.

The objective of the **knowledge acquisition** task is to capture knowledge about some application domain and to represent it in a conceptual schema. The most important problem is the conceptualization problem i.e. to find the concepts best suited for expressing the users requirements.

The **validation** task has the objective of checking whether the conceptual schema specifications are consistent and whether they correctly express the requirements informally stated by the users community.

Let us call Knowledge Engineering the set of knowledge based tasks of requirements engineering including:

(i) **acquisition** of domain dependant knowledge,
(ii) **abstracting** and **conceptualizing** from this knowledge relevant parts of the application domain.

(iii) **generating** conceptual schema elements from previous conceptualization tasks,
(iv) **validating** the conceptual specifications.

Complementary tasks are required to organize and synchronize knowledge based tasks. The term Process Engineering is used to refer to this collection of tasks which comprises:

(i) **guidance** of the conceptual modelling process,
(ii) **explanation** of achieved results,
(iii) **interfacing** with analysts and users.

2.2 FUNCTIONAL ARCHITECTURE OF ALECSI

Alecsi aims at supporting both knowledge engineering and process engineering. This leads to the functional architecture of the tool depicted in figure 2.

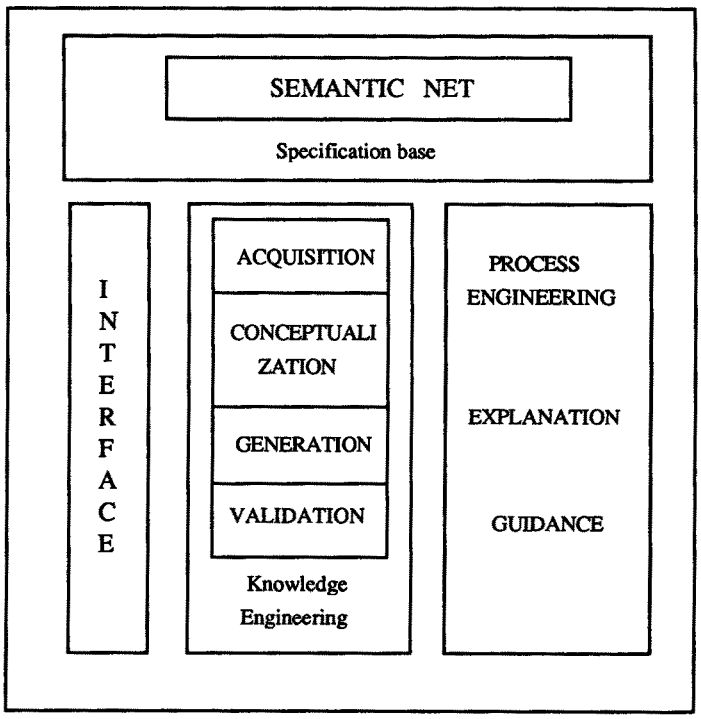


Figure 2 : Alecsi's functional architecture

Knowledge about some application domain can be **acquired** in three different ways:

- the **unstructured** one through natural language sentences,
- the **soft-structured** one based on three basic interconnected concepts (entities, actions and events),
- the **strongly-structured** one corresponding to an object-oriented view of the future IS.

These three levels of knowledge are simultaneously stored in a **semantic net** which corresponds to specifications base (or repository) of the workbench.

In order to help the analyst in the conceptualization problem Alecsi, includes a **modelling support** functionality which maps in a automated-way knowledge from one level of abstraction to the following one (from unstructured to soft-structured and from soft-structured to strongly-structured).

Alecsi tries to avoid replication of work. Using its own expertise in abstracting from descriptions of real facts, Alecsi is able to generate part of a given conceptual schema i.e. to **generate objects** which do not result of the mapping of natural language sentences onto strongly-structured elements of the conceptual schema.

Finally Alecsi includes a **validation** function to check the consistency and completness of the semantic net at the different levels of abstraction. The validation function is coupled with a **correction** function that aims at proposing to any detected anomaly one or several ways for correcting it.

Regarding the **Process Engineering**, Alecsi offers three kinds of help:

- it guides the conceptual modelling process,
- it is allowed to explain why a certain result has been achieved,
- it combines graphical interfaces and natural language for interfacing with the analysts and users.

Requirements engineering is guided by Alecsi as illustrated in figure 3.

Alecsi pilots the construction of the conceptual schema.

The analyst may interact with the semantic net whatever the level of specification he chooses. The tool analyses the specifications status and may react in different ways:

- to inform the analyst of detected inconsistencies, ambiguities or incompletness and to suggest corrections,
- to generate new elements and complete the semantic net,
- to change the semantic net contents in order to map natural language sentences onto more structured elements of the conceptual schema.

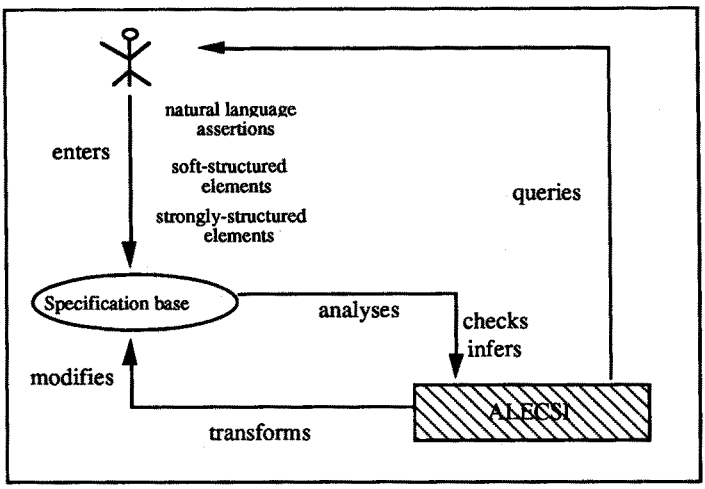


Figure 3 : The guided process engineering of Alecsi

2.3 INTERNAL ARCHITECTURE OF ALECSI

From a system point of view Alecsi must be regarded as an expert system comprising an **inference engine,** a **rule base, a fact base** and **interfaces** (figure 4).

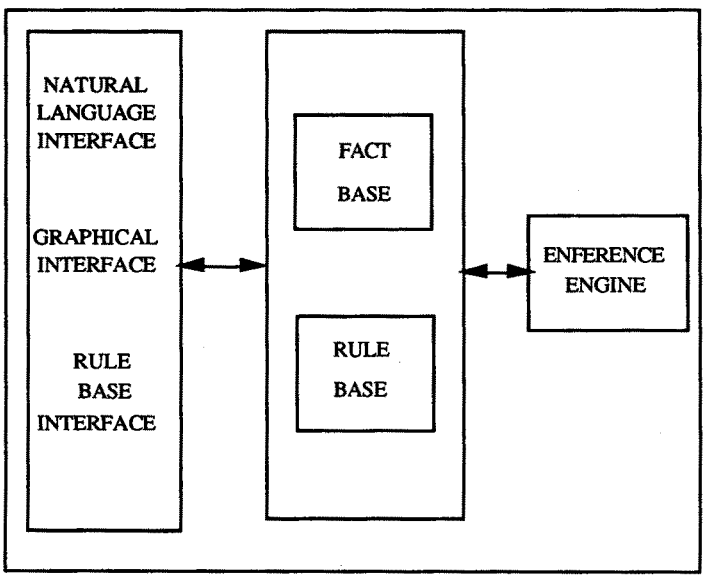


Figure 4 : Alecsi's internal architecture

Alecsi is implemented in PROLOG; thus the **inference engine** is the Prolog compiler.

The **fact base** corresponds to the specification database i.e. the semantic net.

Interfaces are either graphical and natural language interfaces for communicating with the analysts and end-users introduced in figure 2 or the specific Prolog interfaces used by the Alecsi administrator to change the rule base.

The **rule base** is composed of production rules which allow the workbench to support the process engineering and the knowledge engineering as introduced in the previous section. Knowledge engineering tasks and process engineering tasks are automated through this rules. They represent the expert knowledge acquired by the workbench from a number of requirements engineering experiences.

The next section gives a brief overview of the specification base needed to illustrate in section 4 an Alecsi's session.

3. THE SEMANTIC NET

The semantic net integrates three forms of facts illustrated in figures 5, 6 and 7.

Natural languages sentences (figure 5) may be used to capture knowledge about some application domain. This is probably the most natural way to acquire domain-dependant knowledge especially when interviewing people working with the real application. Sentences may describe real facts (assertion 1), static or dynamic constraints (assertion 2) or rules that govern the application life (assertion 3).

Assertion_1 : The subscribers request for loans
Assertion_2 : A subscriber is not allowed to loan more than three books at the same time
Assertion_3 : A loan is accepted only if a copy is available

Figure 5 : Assertions in natural language

The experienced analyst may be able to conceptualize relevant parts of the application domain and to capture knowledge in an already more structured and typed form. Thus, he can enter the specifications through the

graphical interface as a net of concepts. For example in figure 6 the "BOOK" is described as having a title, a reference, authors, several copies and be possibly requested for loan. This **soft-structured** way of knowledge representation allowed three types of nodes (entity, action and event) and stamped binary relationships among them represented by non-typed arcs.

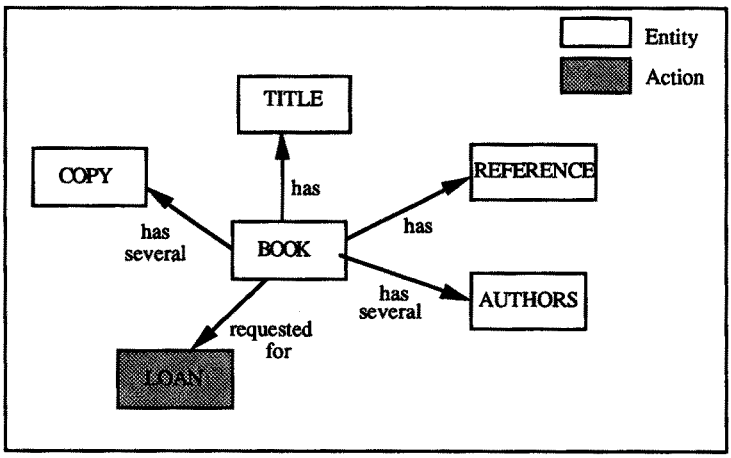


Figure 6 : Soft-structured representation

Finally, elements of the conceptual schema are **strongly-structured** objects. An object has a static part and a dynamic one. The structure of an object is based on the "part-of" and "member-of" links. The behaviour of an object includes actions (which change its state) and events (particular state changes which activate actions on other objects).
Objects may inherit from others objects. In addition, their definition is constrained. For instance any object must have an identifier and at least one associated action.
Objects of a given conceptual schema are described through typed nodes (entity, domain, action, event and constraint) and typed arcs (part-of, is-a, member-of, modifies, triggers and constraints) of the semantic net.

For example in figure 7 "TITLE", "REFERENCE" are part-of the definition of the conceptual object "BOOK" as well as its set-of "AUTHORS" and its set-of "COPY". The book is created by the action "CHECKING" and deleted by the action "GET AWAY". Similarly a copy of a book is an object composed of a "STATUS" and a "COPY_NUMBER". A copy may be loaned and this changes its "STATUS" (the "LOAN" action).

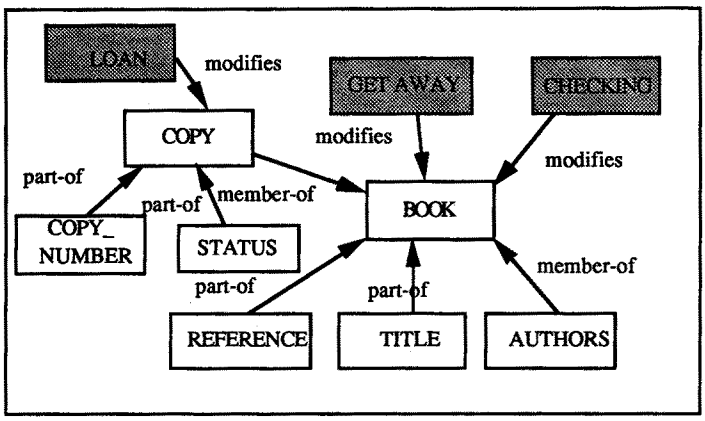


Figure 7 : Strongly-structured representation

The three forms of knowledge can coexist in the semantic net. At any time of the requirements engineering phase the analyst can enter a new element in any of the three forms. In addition he can directly, through the graphical interface, change the semantic net elements whatever their level of modelling.

Alecsi supports the analyst to abstract one level of knowledge to another one. It helps him to progressively conceptualize relevant parts of the application domain and reach a precise and structured description of the collection of conceptual objects. This will be examplified in the next section.

4. ILLUSTRATION OF AN ALECSI SESSION

The scenario presented in this section aims at illustrating the requirements engineering support provided by Alecsi. For clarity reasons, the presentation is based on simple examples which are extracted from a real case study of an automated subscription library. The figures 9, 10, 11, 12, 13, 14, 15 and 16 which are presented in this section, are hard-copies of Alecsi screen.

INITIAL STATEMENTS

Let assume that the analyst starts with the following natural language assertions.

> *The subscribers request for loans. A subscriber has a numero, a name and an address. When a subscriber requests for a loan, the request is accepted if a copy is available, otherwise, the request is put on a waiting list.*

Figure 8 : Natural language assertions for the library case study

INTERPRETATION

By selecting the menu line "Interpretation", the analyst triggers the automatic analysis and interpretation of the natural language assertions and their mapping onto a net of concepts. The result of this task is shown in figure 9 for initial natural language assertions of the library case study.

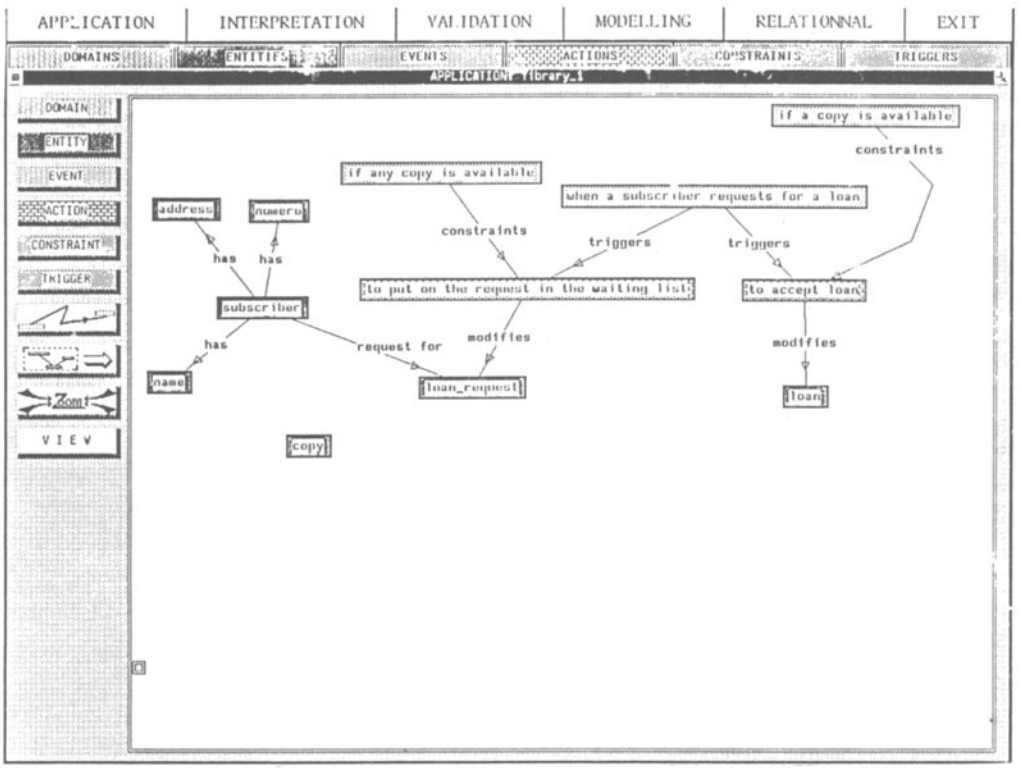


Figure 9 : Result of interpretation

The interpretation of natural language assertions is one of the conceptualization support provided by Alecsi. The interpretation task is based on a linguistic approach borrowed from the Fillmore theory "case for case" [FILL. 68]. The semantic interpretation of natural language sentences (see [PROI.89] for a detailed description) is based upon their grammatical structure and the meaning of verbs.

From this interpretation the tool infers:

- **nodes of the semantic net** of one of the three pre-defined types (entity, event and action). For example SUBSCRIBER is recognized as an entity and "TO_ACCEPT_LOAN" as an action.

- **semantic links** between nodes. For instance , the link defined between SUBSCRIBER and LOAN_REQUEST expresses a relationship between the two entities; the link between the event "When a subscriber requests for a loan" and the action "to accept the loan" expresses that when the event occurred, the action is triggered with the condition "if a copy is available".

ADDITION TO THE SEMANTIC NET

The graphical representation of the generated semantic net may suggest to the analyst some completion of the net.

Following our scenario, figure 10 shows that the analyst has added a new entity "BOOK" and completed the description of the entity "LOAN_REQUEST".

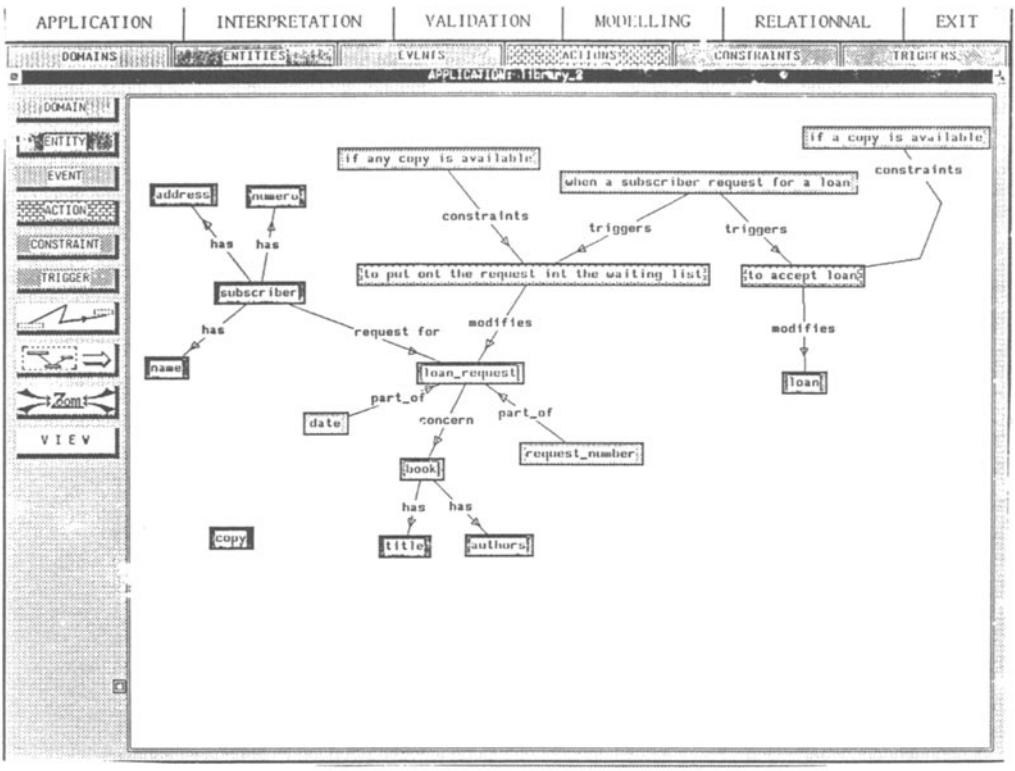


Figure 10 : Completion of the net

The first addition is based on soft-structured links while "DATE" and "REQUEST_NUMBER" are introduced as "part-of" the object LOAN_REQUEST. The soft-structured link between

"LOAN_REQUEST" and "BOOK" remains to be precised. The analyst can mix soft-typed constructs and strongly-structured constructs. As indicated by the menu (figure 10), the graphical editor of Alecsi provides facilities to manipulate the semantic net especially for adding and deleting nodes and links mixing if useful, soft and strong-structured nodes and arcs.

VALIDATION SUPPORT

At any moment, the analyst can request the tool to support him in validating the achieved result.

Validation consists of both the detection of errors and suggestions for correcting them. Thus, a validation session is interactive and often leads to important modifications of the semantic net.

For instance the net presented in figure 11 may result of an interactive validation phase activated on the net of figure 10.

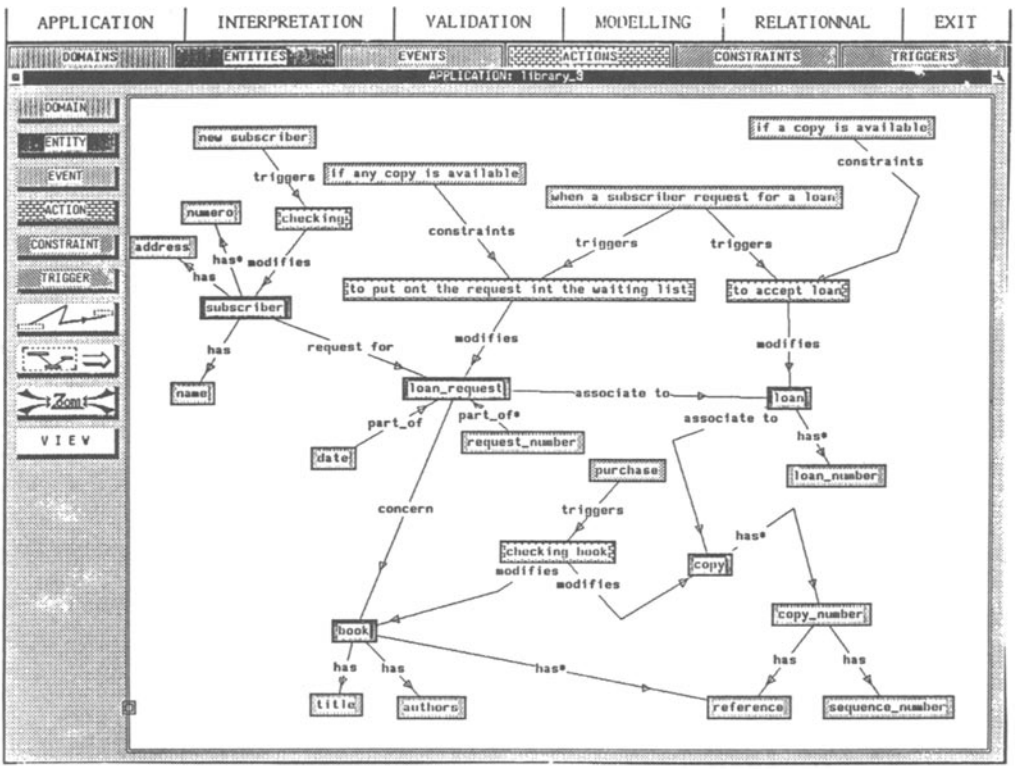


Figure 11 : Result of validation

Modifications correspond to:

- typing of entity nodes into domain nodes,
- connexion of any entity node to an identifier(the connexion is represented by a labelled link with a star),
- integration of the isolated entity node "COPY",
- connection of any entity node to an action node,
- connection of any action node to an event node.

Each transformation results of a dialogue with the analyst. To illustrate this point let us consider the following situation:

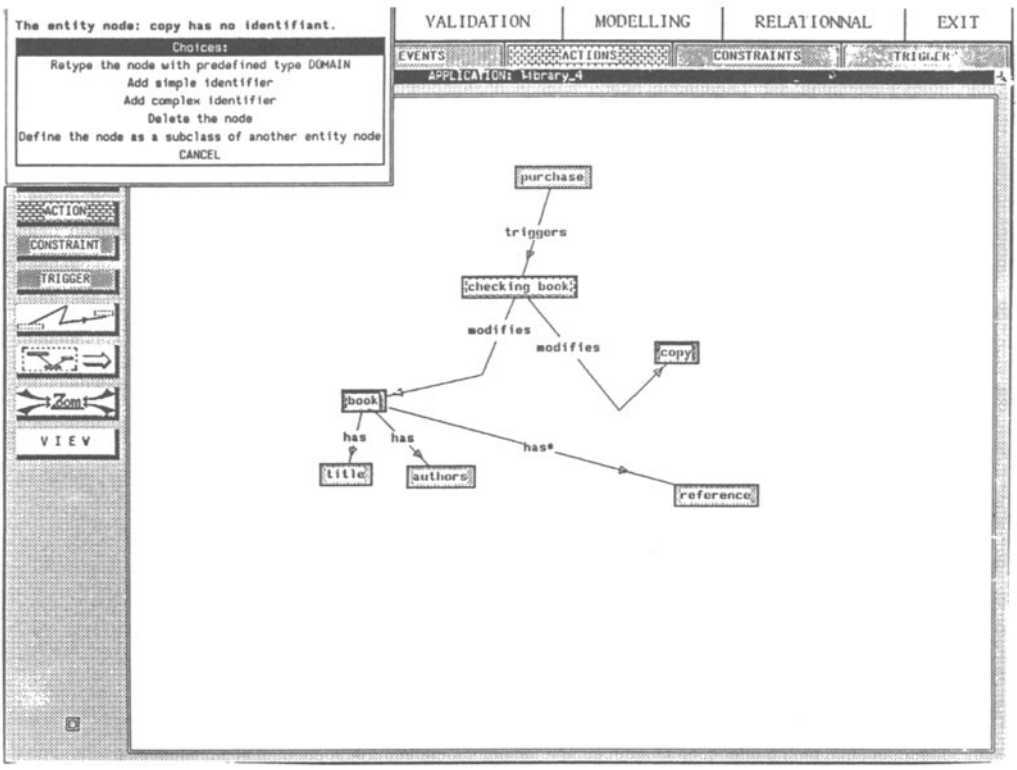


Figure 12 : Incorrect situation

BOOK is identified by its reference while COPY is not. In order to satisfy the constraint that any object has an identifier, Alecsi proposes a collection of possible corrections:

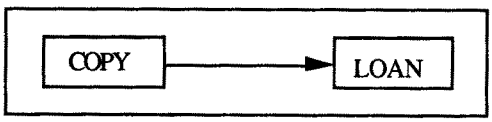
- re-type the COPY node with the pre-defined type DOMAIN,
- add a simple identifier,
- add a complex identifier,
- delete the node.
- define the node as a subclass of another entity node.

As shown in figure 11, the analyst has chosen to add a complex identifier composed of the book reference plus an internal sequence number of copies of a given book.

GENERATING SOLUTIONS

We have already shown that Alecsi supports the analyst in modelling the semantic content of natural language assertions i.e. to map unstructured statements on to soft-structured nodes and links of the semantic net. Similarly, Alecsi operates on soft-structured elements of the net to generate strongly-structured objects of the conceptual schema.

For instance, Alecsi will propose to help the designer in transforming a soft-structured link between two nodes such as LOAN and COPY.



It proceeds in an interactive way, querying eventually the analyst to acquire some needed information. For example the tool requires in the previous case characterization of the link (and its reverse) by three properties.

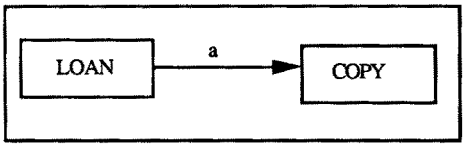
The link <COPY , LOAN> is :

- partial, a copy may be borrowed or not
- simple, a copy may relate to only one loan at a given time
- variable, the same copy may correspond to different loans at different points of time.

Similarly, the link <LOAN, COPY> is total, simple and permanent.

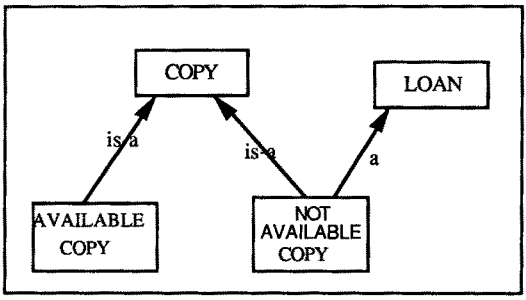
Thus, using its own knowledge, the tool proposes different patterns of structuration. The two that are proposed for the <LOAN, COPY> example are the following.

(1) suggests to type the link as a "reference" between the two object LOAN and COPY with an associated referential constraint (the loan can not refer to an non-existing copy of a book).



(1)

(2) proposes to introduce two sub-classes of COPY: AVAILABLE-COPY and NOT-AVAILABLE-COPY which refers to the LOAN.



(2)

In the figure 13 , the analyst has chosen proposal (2).

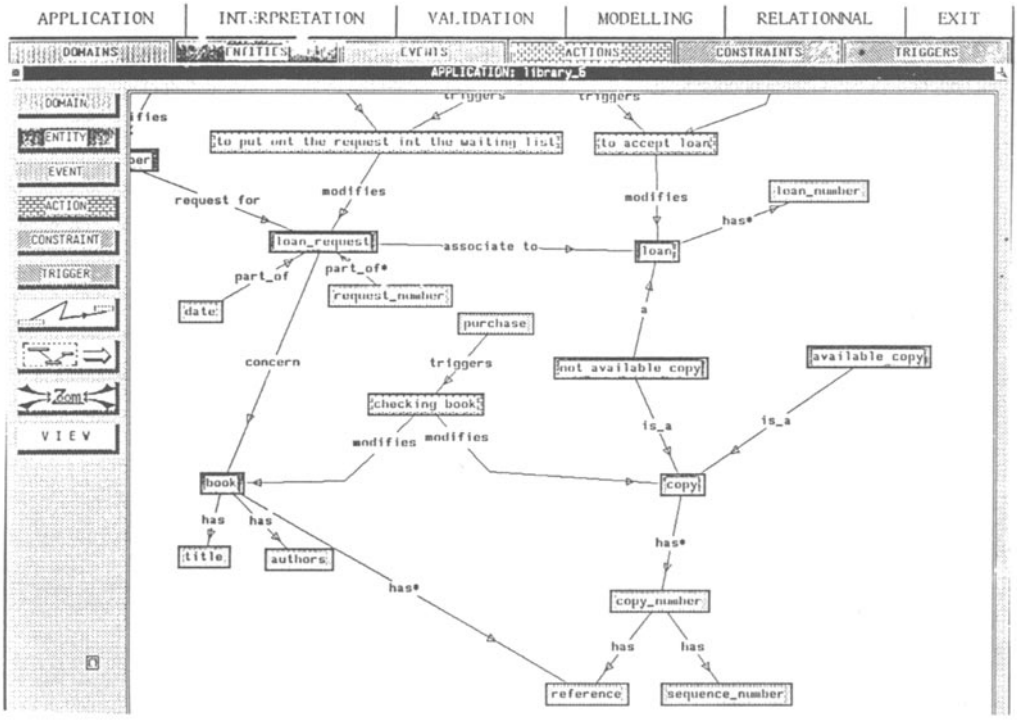


Figure 13

GUIDING THE REQUIREMENTS ENGINEERING PROCESS

Using the menu line, the analyst can select a work session on a specific task as modelling, generation,... Alecsi will often react by triggering the session but may suggest others as pre-requisites for the selected one.

Let's assume, for example, that starting with the net presented in figure 14, the analyst requests for a modelling session.

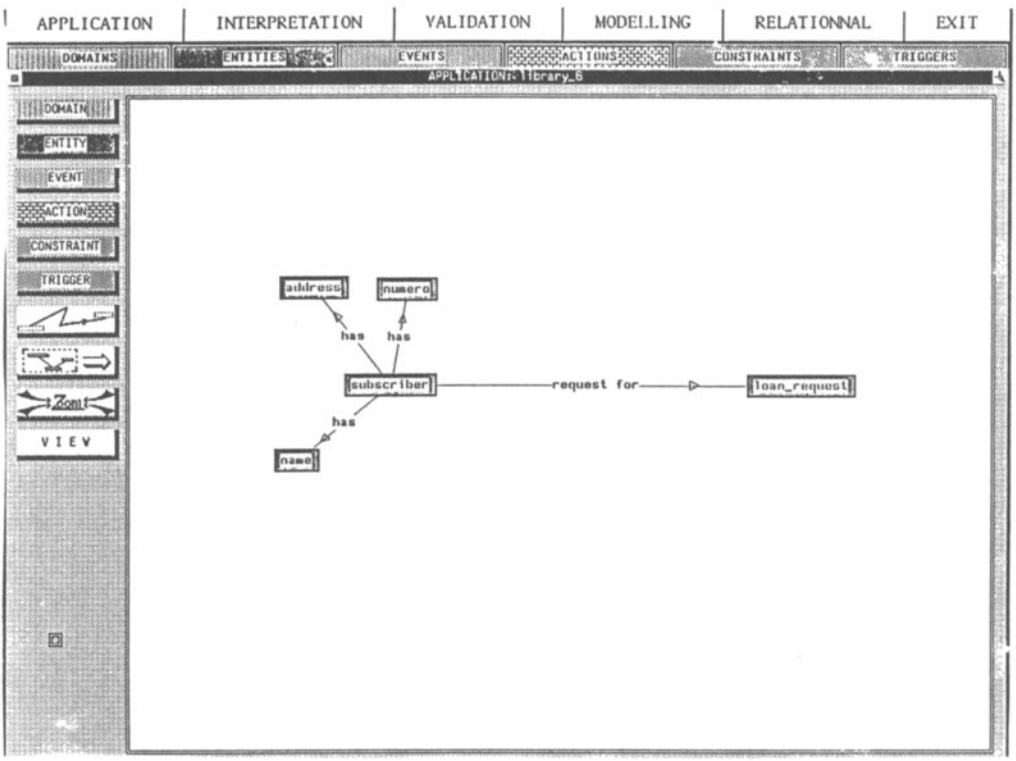


Figure 14

Thus Alecsi will guide the analyst to a validation session in order to take benefits of the support provided by the tool when validating an incomplete specification.

Specification will lead to precise the two nodes SUBSCRIBER and LOAN-REQUEST by adding identifiers, actions and events (figure 15). Then, Alecsi identifying the pattern <Entity - Relationship - Entity> will be able to guide the analyst in its conceptualization.

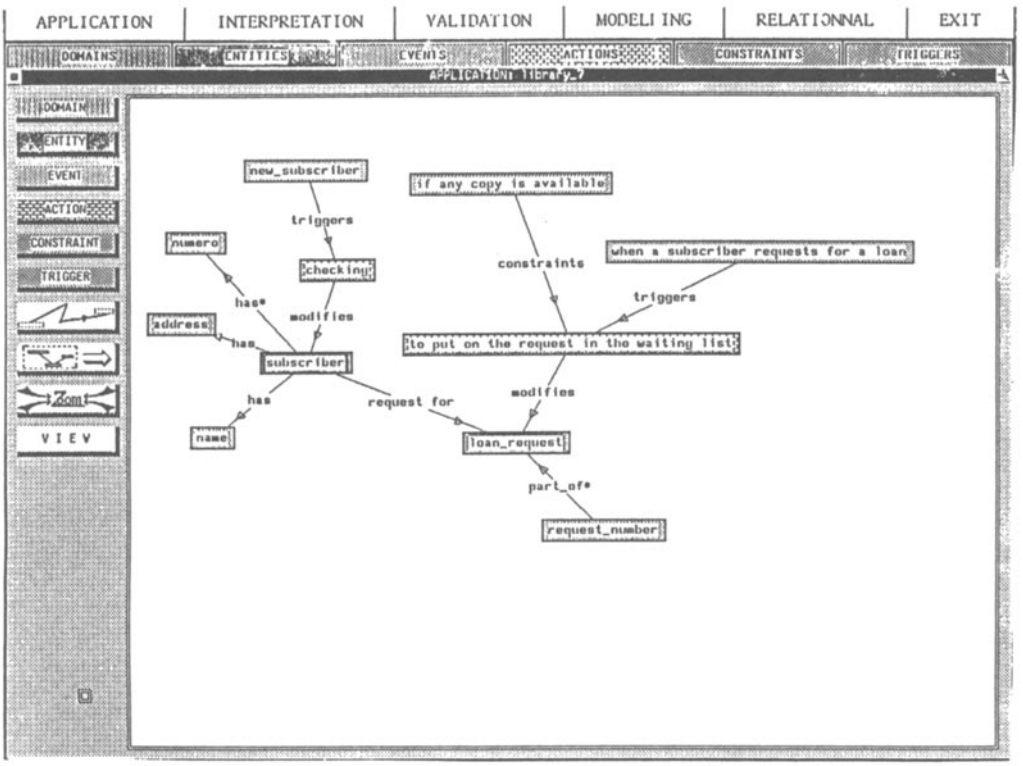


Figure 15

NETWORK SCANNING

Alecsi provides facilities for scanning the semantic net. A kind of graphical querying language is implemented. This mechanism allows to define a query using some menus and shows the answer in a graphical sub_window named a view. This sub_window has the same behaviour than the main one. So it is possible to manipulate the net in this view as in the main graphical window (the model-view-controler paradigm is used for this).

The figure 16 shows the answer of the following query applyed to the net of the figure 11 : "*Show all connected entities to LOAN_REQUEST and its actions*".

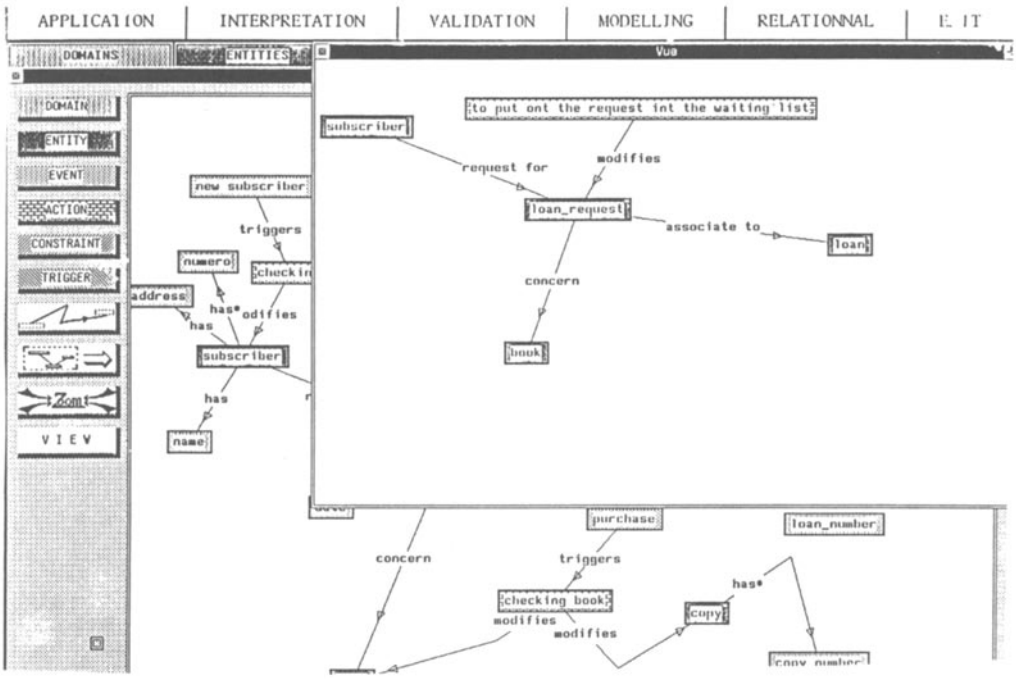


Figure 16

5. CONCLUSION

Alecsi is a first approach to realize a tool which aims at supporting not only the schema description and the schema documentation but also the design process leading to the schema. Alecsi focuses on supporting the designer in the conceptual modelling process i.e. the requirements engineering process of constructing the specification by abstracting and conceptualizing relevant parts of the application domain.

Nowadays, Alecsi is an operational prototype developed in C_prolog (Base of rules, about 15,000 lines) and Objective_C (Interface, about 12,000 lines).

The further works around the actual prototype are :
- to integrate a full object oriented approach both in the underlying model and the method
- to add some reusability mechanisms in order to support more efficiently the user during the requirements analysis phase,
- to increase the formalization of the conceptual modelling process in order to automatize as far as possible parts of the analysis and design processes.

VI. REFERENCES

[BROD.82]: M.L. BRODIE, E. SILVA : "Active and Passive Component Modelling: ACM / PCM", in Information Systems Design Methodologies : a Comparative Review, Ed. Olle T., North-Holland, 1982.

[DUBO. 89]: E. DUBOIS : "A logic of action for supporting goal-oriented elaborations of requirements" in Proc. fith international workshop on software specification and design, 1989, pp160-168.

[FILL. 68]: CJ. FILLMORE : "The Case for Case", in universals in linguistics theory: Holt, Rinehart and Winston, Inc., E. Bach/RT. Harms (eds) 1968.

[GUST.82]: M.R. GUSTAFSON, T. KARLSSON, J.A. BUBENKO : "A Declarative Approach to Conceptual Modelling", in Information Systems Design Methodologies : a Comparative Review, Ed. Olle T., North-Holland, 1982.

[HAGE. 88]: J. HAGELSTEIN, E. DUBOIS, A. RIFAUT :"Formal requirements engineering with ERAE", Philips Journal Of research, vol 43, 3/4, 1988, pp 393-414.

[HAMM.81]: M.M. HAMMER, D.J. McLEOD : "Database Description with SDM: A Semantic Database Model", ACM Trans. on Database Systems, Vol 6, N^{O} 3, pp 357 - 386, 1981.

[MYLO.80]: J. MYLOPOULOS, H.K.T. WONG : "Some features of the TAXIS model", in proc. of 6th Int. Conf. on VLDB 1980.

[PROI. 89]: C. PROIX : "OICSI un outil d'aide à la conception des systèmes d'information : spécification et réalisation", thèse de doctorat de l'université Paris 6, december 1989.

[ROLL. 79]: C. ROLLAND, S. LEIFERT, C. RICHARD : "Tools for information dynamics mangement", Proc 5 th Conference on VLDB, Rio de Janeiro, 1979.

[SMIT.77]: J.M. SMITH, D.C.P. SMITH : "Database Abstraction : Aggregation and Generalization", ACM Trans. on Database Systems, Vol 2, N^{O} 2, 1977.