17

AMI: A Case Tool based on Artificial Intelligence Techniques for an Interactive Modelling for Analysis.

L. LASOUDRIS[a] and M. MAURICE-DEMOURIOUX[a,b] and R. VOYER[c]

[a]Institut National des Télécommunications 9, rue charles Fourier - 91011 EVRY - FRANCE
E-mail: lasoudri@galaxie.int-evry.fr
[b]LAMSADE - Université Paris IX Place du Maréchal de Lathre de Tassigny - 75775 Paris Cedex 16 - FRANCE
[c]LAFORIA - IBP - U. Paris VI Tour 46-00 - 2ème étage - 4, place Jussieu - 75252 Paris Cedex 05 - FRANCE

Abstract

Today, CASE (Computer Aided Software Engineering) tools functionalities give some help in data and process description (mainly in managing the results of analysis). They do not help with efficiency in analysis and design process. Our model brings some aid in O.M.T. (Object Modelling Techniques from Rumbaugh) modelling processing.

AMI uses: Model examples (schema) base, Pattern examples (generic modelling structures of business area) and Generic business area knowledge. AMI allows software analyst or designer both to refine description concepts which is a schema part beeing in progress, and to extend concepts by interpretation of a business area words lattice. AMI implementation uses an A.I. technique: Case-Based-Reasoning (CBR). CBR is very interesting by increasing ability in reasoning activity. In CBR system searches of cases are mainly based on terminology indexing. Indexation system does not allow sufficiently semantical expression. In our area we need more. So we choose for AMI a conceptual graph. In fact, subsumption links and matching possibilities between conceptual graphs allow us to compare different abstraction levels and to realize selective searches for graph verifying some properties.

Today, the implementation of this tool with Smalltalk-80 and its environment VisualWorks is in progress.

Keyword Codes: D.2.1; D.2.2; I.2.4
Keywords: Software, Software Engineering, Requirements Specifications; Tools and Techniques; Computing Methodologies, Artificial Intelligence, Knowledge Representation

1. INTRODUCTION

Modelling activity is the main analysis task in software engineering. Efficiency and quality in modelling activity are two principal criterias for software engineering analysis activity. Using knowledge bringing by previous stored models, works efficiently in modelling tasks. It consists of being able to reuse models or model extracts and to adapt them if it is necessary to satisfy a new analyst need. But, in most cases, it is very difficult to realize a matching between needs expression and previous models. Nowadays in this area, CASE tools "never solve" this problem. So this problem must be solved by the analyst. In this paper, we propose an interactive resolution method using Artificial Intelligence techniques and methods which helps analyst.

In a first part, we present the notion of model as it is used in analysis phase. In a second part, we present modelling process problem in this phase and its limits. We present the main tasks which make problem for the analyst. Then we will show reuse defined in a tool is a good solution to solve this problem. But reuse alone is not sufficient, it must be completed by a concepts emergence techniques. So, we note the state of art in first tools and in second in researches. The last part, before conclusion, presents a tool named AMI, which is in progress in our laboratory.

2.MODEL NOTION

Software life cycle could be decomposed in three phases: requirements/analysis, design and implementation.

M. Lee et al. (eds.), *Software Quality and Productivity*

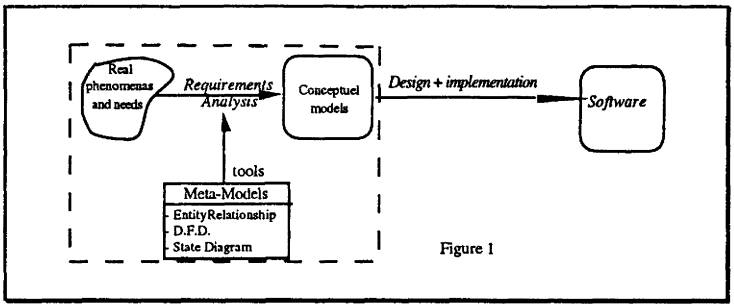


Figure 1

Our paper is relative to requirements/analysis phase (this part corresponds to concepts included in dotted line rectangle, cf. figure 1.).

Software engineering uses model during all phases of its life cycle especially in analysis requirement phase. A model is a particular aspect conceptualization of a system (data, function, behavior). In litterature, the term of model (or diagram, schemata) is used indifferently to designate modelling language or object of this modelization. In this paper, we will use "meta-model" to designate the language and "model" to designate the result of the analysis phase.

Using meta-model leads to a better comprehension of requirements/analysis problem. More, modelling task increases communication between project actors.

Software engineering methods like O.M.T. [1], O.O.A [2], MERISE [3] or OOSA [4, 5] propose specific models, which could be classify according to three types: Entity/Relationship, Data Flow Diagram (D.F.D.) and State Diagram.

In this paper, we will use Object Modelling Techniques (OMT) to illustrate our ideas. For example, OMT suggests to use:

- An "information model", which looks like an extended entity/relationship model with encapsulation of object behavior.
- A classical data flow diagram which illustrates system works.
- A state diagram which describes for a class, possible states and transition states for objects.

3. MODELLING PROCESS PROBLEM IN ANALYSIS PHASE

Existing methods do not well helping during the analyst process. In fact, the reasoning by which we abstract requirements expression to give rise to relevant elements is not well known and not often formalized.

Modelling process is divided, in a company, in several complementary subsets of projects. This leads to analysis models which could be semantically redundant in an unwishing manner (homonymy, synonymy, ...). Company must assure and preserve coherence, non redundancy, and utility of modelized knowledge, especially during new models creation. For these reasons, we propose to use a reuse function.

3.1. It Is a Must To Help Model Components Emergence Process

Modelling process is creative. It is an expert work. Methods using specific meta-model give concepts which must be modelized, and the associated formalism. Nevertheless, they indicate just a little how to realize concepts emergence.

We think that it is necessary:

- to better understand concepts emergence process in requirements/analysis phase
- to know what are indispensable knowledges to realize this emergence.
- to know how this knowledge should be formalized.

We think that this type of formalization is necessary for developping active and intelligent software tools. These tools must be able to help analyst in concepts emergence process.

Today, CASE (Computer Aided Software Engineering) Tools are only able to manage models. They help in model entry using graphic interfaces, storing them into databases, validating and documenting them. They do not help (or a little) in the modelling process because they do not well reuse previous models.

3.2. It Is a Must To Reuse Existing Models or Components of Existing Models

Company view: Set of knowledge bringing by different types of models (data, function, behavior) is the memory structure of a company.

Semantic wealth and precision expressed by models allow a good understanding and communication between actors in a company and consequently reuse of model components is facilitated. In fact, there is an overlapping betwwen concepts using in different models.

Company wishes that coherence, non redundancy, relevance of modelling knowledge are ensured, in particular when new models are elaborated. Consequently, we have to propose an efficiency reuse function which allows analyst to use an adapted structure for new redefined knowledge.

Analyst view: Analyst wishes to work with an existing model then adapts it. He works using existing patterns or model examples and transforms them to reached his specific needs. Then, he gains time in modelling tasks.

4. REUSE

Our goal is to help analyst in his tasks which make problems for him [6].

The first task for which it seems to be helpful to bring an aid is the models storing and researching. This function requires a tool which allows reuse.

Modelization art is complex, analyst needs are very difficult to define. It is also an area in which creativity is very important. So,it seems to be judicious that our tool allows reuse, brings also an help in emergence and definition of analyst needs.

4.1. A Tool Allowing Reuse

Making reuse facility requires to have got a models base, a searching module for picking models or models components which are exactly or closely a response to specific needs for the new model to be built. This latter case requires an adaptation module.

We have to integrate new models (and knowledge they are bringing with) in existing models bases. We must do this integration without forgotten this simple idea "they will be reused later".

These indispensable functionnalities in a tool which allowed reusing models require a high semantic content (specific knowledge). In fact, this tool must have two types of knowledge.

1. Knowledge on existing models components, on reference area, on used meta-model.
2. Knowledge which makes easier models search and capture and which facilitates their adaptation according to expressed needs.

We are seemed to determine necessary knowledge to realize these functionnalities, and in which manner this knowledge should be formalized and manipulated.

4.2. A Tool Which Facilitate Concepts Emergence

Tool must allowed needs expression and understanding. Existing methods do not work totally for the modelling process. Through a specific meta-model, they indicate concepts to be emerged. In fact the reasoning by which people is able to abstract from needs expression the necessary model elements is unwellknown and few or not formalized. Modelling process requires creativity and expertise.

It seems us that it is necessary to have a better understanding of concepts emergence process in analysis phase, to know what is the necessary knowledge which allowed this emergence and in which manner this knowledge should be formalized.

We think that this formalization is necessary for intelligent active software tools development. These tools will be able to help the analyst in concepts emergence process

Help could be bring on two levels:

First level. Analyst must be allowed to formulate information needs (model or component model) by the mean of a query. As far as possible, analyst must express his need with its own terms (its own words) and its own structure (semantic links between concepts of his query). Tool must detect ambiguities into the query, then it must help analyst, according to his wishes, to remove these ambiguities.

Moreover if there is any model in the base which do not satisfy sufficiently the query, the tool could used knowledge about business area to reformulate initial query in terms closed to initial terms (synonym, specialized, generalized, analogous terms, evocated ideas, connected terms, ...).

Second level. Results produced are model extracts which satisfy the expressed needs. Moreover, the obtention of the environment (in model terms) of the results allows an analyst increased thinking and new ideas emergence.

As far as we know, concepts emergence as it is presented in this paper is not integrated into present solutions or tools. There is only some reuse which was made in use.

5. PRESENT TOOLS FOR MODELLING HELP

Present CASE Tools are models manager: they help in getting models using useful graphics interfaces, in storing them in a database, in validating and documenting them They do not well work in producing models because they do not sufficiently integrated necessary knowledge about modelsl extracts emergence.

Present CASE query languages [ERQL (Entity/Relationship Query Langugaes: Gamma International) or GQL (GraphTalk Query Language: Parralax)] allow access to data in repository. The repository is structured on a meta-model composed with segments which concepts of a specific method (Merise [3]) or concepts built before by analyst. Repository is instantiated with model datas described by analyst.

Query languages allow to structure the query around variables defined in meta-model and profit of the advantage of standard functions like count, min, max and so on. Query variables could be parametrized. GraphTalk (Parallax) furnishes some predefined functions which could be utilized in C programs interfacing the GraphTalk tool (links between objects list, model objects list, ...).

The advantages of these tools

These tools allow us to obtain existing models extracts. So it is possible for the analyst to reuse them, and to document applications.

The drawbacks of these tools

It is necessary to well know the model structure to obtain a model extract. In some cases, people must know value of variables to find instantiated concept meta-model.

There is no possibility to look for a component with neither synonym nor evocation links. In others words, search may be efficiency only if the analyst well-knows existing stored models. Queries give model extracts in textual form and not in the form which is associated with the specific meta-model.

6. PRESENT RESEARCH IN MODELLING HELP

Research works show that models reuse process requires to abstract and to structure knowledge stored into them.

The principle is: first take account of model construction process is only possible if modelling tasks and scheduling of process are formalized. This requires that modelling tasks could lie on generic modelling structure. These structures must be independant of specific applications, so they could be reused in many modelling tasks. In others words, their assumptions are that there are many phenomenas which could be modelized by identical structures. Specific model could be built by instanciation of generic structures [7, 8].

The drawback of this approach is the necessity to a-priori build a set of knowledges (generic structures) allowing a large covering of analysis modelling tasks. This approach seems to be difficult taking account of the great number of possible generic structures and the difficulty to find them (choice among abstraction levels and classication criterias).

7. MAIN CARACTERISTICS OF "AMI"

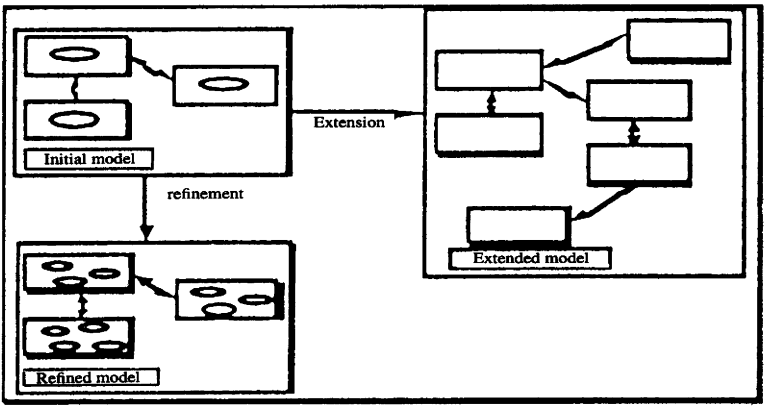
AMI system is a tool which brings help in business application analysis. It is based on the reuse notion of models examples, models patterns, and domain predefined generic knowledge.

An example is a previous existing and validating model (OMT model) or model extract.

A pattern is a stable instanciable generic modelling structure. This concept is based on concepts like Frames [9].

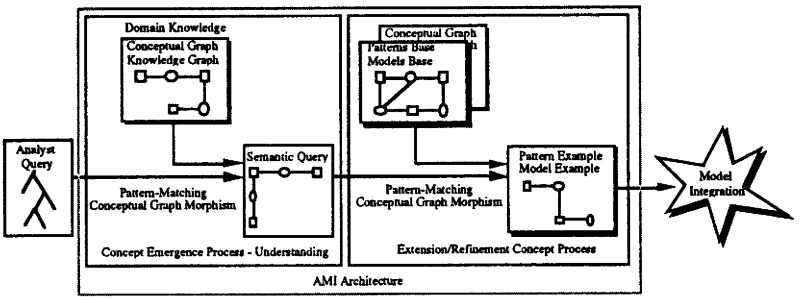
Domain generic knowledges are formalized in a terms conceptual graph (which is named domain graph in the continuation of the paper), which use allows concepts emergence as it is presented in 4-2 paragraph. Links typology induces a specific semantic interpretation. Conceptual graphs are initially developped to represent semantic expressed by natural language. Based on semantic nets, they are used for concepts representation like object, action, activity, event, links (between these OMT concepts). In a formal point of view, a conceptual graph is a bipart graph composed with two types of nodes: concepts nodes and relationships nodes.

AMI helps analyst in refining or extending model part using domain graph interpretation.



Today, the implementation of this tool with Smalltalk-80 and its environment VisualWorks [10] is in progress.

7.1. Logical Architecture



Make use of modelling help is based on case based reasoning (CBR) technics ([11], [6], [12]) and information search [13] and it is done with two phases:

Understanding: It is realized by a flexible, interactive, gradual search in such manner that analyst need formulation progresses step by step. Specified need may be reformulated in interactive manner using knowledge graph. The goal is to obtain a non ambiguous semantic query according to analyst real need.

We could distinguish in conceptual graph several links types: generalization/specialization links, synonymous links, ideas evocation links or closed terms (group terms). Knowledge graph structure is based on the idea of thesaurus in documentation databases area (*Roget's International Thesauraus* and *Webster's 9th New Collegiate Dictionary*).

Matching: Model construction induces an interpretation of its meaning. Matching allowed to select models (or model extracts) not only exclusively those which respond exactly to need expression but also those closed with needs expression.

Matching modules try to satisfy analyst needs just as it is expressed in query, then to compare it with previous stored models examples or patterns.

Matching module quality is associated to model internal representation techniques, to query expression quality, and to matching methods between these two knowledge types.

These latters use indexing and matching functions. They also use strategies: for examples: what we have to do for reducing solutions number if it is necessary, or if we do not find any solutions. Must we have to explicitly and/or implicitly reformulate the query or to build a more generic query ?

By reconstruction of Conceptual Graphs (CG) [14] in terms of more basic "knowledge graphs" [15], we use standard pattern matching techniques [16] for implementing several aspects of the CG theory. In fact, the fundamental CG projection, which determines the specialization/generalization relations between CG, is expressed in terms of pattern matching. So, we can easily modified and/or extended our model since its implementation is fully described by patterns [17]. As a result, this approach improves both the quality of the concept emergence process and the efficiency of the retrieving of model examples and pattern examples.

7.2. Scenarios For Using AMI

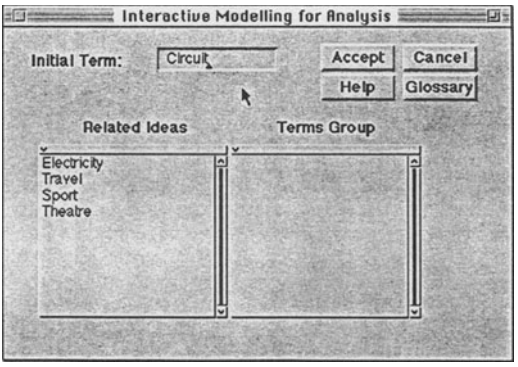
It is possible to use AMI at different conception levels. It helps analyst in two tasks. We describe for each of them the using mode of AMI

7.2.1. Help For Definition Concept Phase (refinement)

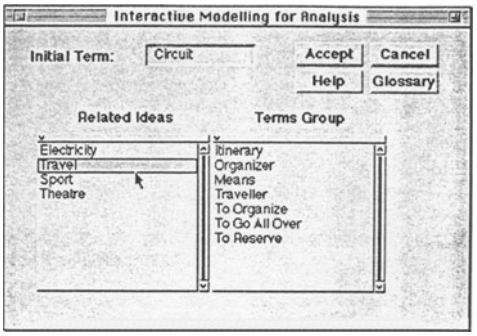
This phase is composed with four steps which are strictly sequential:

a. **Process Initialization.** This step aims to collect and structure initial term from which analyst wishes an help. Recognition is lexical recognition with a terms dictionnary.

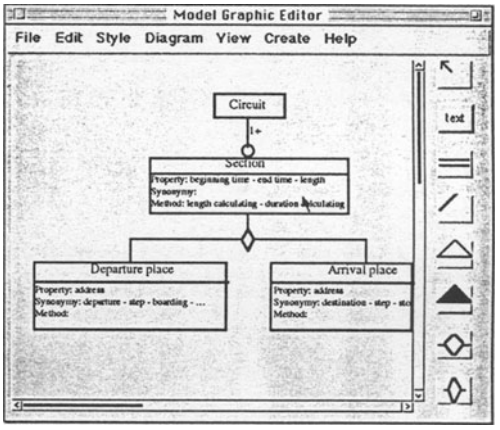
b. **Related Ideas Evocation.** From a recognized term, AMI, using domain terms stored in conceptual graph, proposes main entities closed directly to initial term. We name this entities: ideas. Analyst select one main idea among the proposed ideas.



c. **Main Ideas Selection.** Once a main idea is selected, AMI proposes a set of terms synonymous of the initial term.



d. **Concept description.** Once a key concept is selected, AMI proposes a pattern (structural description) and/or examples (models extracts of different nature : data, behaviour, functional).



In accordance with selected principles, ideas and concepts evocation phases may be more or less complicated. In fact, it is simple terms equality or at the contrary conceptual graph morphisms.

7.2.2. Help For Concept Extension Phase

From same structure and same conceptual graph, the concept extension phase is equivalent to concept refinement phase described before.

This phase is composed with four steps which are strictly sequential:

a. **Process initialization. (ie 7.2.1)**

b. **Related ideas evocation.(ie 7.2.1)**

c. **Annex concepts evocation.** Once a main idea is selected, AMI proposes a set of group terms obtained from the main idea selected. A group term is obtained using a link from term idea. Then analyst selects the link he wishes to explore.

d. **Concept extension.** Once a key concept is selected, AMI goes through process and sets analyst in concept definition phase.

Now, we have to explicite a scenario using a leisure area example.

7.3. Scenario Example: Help For Concept Extension Phase

Process initialization: Analyst modelizes the concept of "circuit" and looks for defining its modelling environment.

Related ideas evocation: Ideas evocation is done by searching through domain graph.

Annex concepts evocation: From ideas evocation links AMI delivers groups-terms.

Concept extension: From "organize and program" group term AMI proposes to analyst one of the terms of the groups. So it delivers as a result a set of examples and/or a pattern representing the selected term if it is possible. If not, AMI proposes one or several examples and/or pattern relative to others terms of the same group terms.

8. CONCLUSION

In software "requirements/analysis" process, recent tools do not make easy and efficiency previous models reuse. Our tool AMI based on Case Based Reasoning, Production System, Conceptual Graph and Semantic pattern make easy and efficiency Reuse in analysis phase. AMI allows analyst to reuse previous models, or to reuse stable pattern delivered from an analyst query. These queries are matching, by using a production system, to conceptual graph which represents associated knowledge domain. This knowledge is very efficiency because it uses not only synonymous terms but also "generalized/specialized" and analogous terms, evocated ideas, evocated terms. So AMI is able to deliver model or model extract after refinement or extension from an initial term that analyst tries to modelize.

Our future works is both to increase pattern semantic and to adjust thesaurus represented by conceptual graph. Others works should be desirable like different points of view on models, knowledges, coherence, integration of views and so on.

REFERENCES

1. RUMBAUGH Object modeling techniques Prentice Hall 1991
2. COAD and YOURDON, Object-Oriented Analysis ; Prentice Hall International - 1991
3. TARDIEU, ROCHEFELD, COLLETI, (in French) MERISE, Démarche et pratiques ; Edition d'Organisation - 1985
4. SCHLAER and MELLOR, Modelling the world with data ; Yourdon Press - 1991
5. SCHLAER and MELLOR, Modelling the world with states ; Yourdon Press - 1992
6. KOLODNER, Improving human decision making through case-based decision aiding AI Magazine, Vol 12, n°2
7. GROSZ, (in French) Formalisation des connaissances réutilisables pour la conception de systèmes d'information, Thèse de doctorat de l'université de Paris VI - 1991
8. NGUYEN, Un catalogue de type de relation: Une approche sémantique pour la conception dans un formalisme entité-relation étendu, Congrès INFORSID 91 - Paris.
9. MINSKY, A Framework for Representing Knowledge, The Psychology of Computer Vision, pp211-277 ; P. H. Winston Ed., Mc GrawHill - 1975
10. PARCPLACE. "VisualWorks: Reviewer's Guide - release 1". ParcPlace Systems, Inc. 1992.
11. BAREISS, Exemplar-Based Knowledge Acquisition : a Unified Approach to Concept Representation, Classification and Learning ; Boston : Academic
12. SLADE, Case-Based Reasonning : a Research Paradigm ; AI Magazine 12(1) : 42-55
13. Van RIJSBERGEN, A new theorical framework for Information Retrieval, proc. of the ACM Conference on Research and Development in Information Retrieval ; Pisa, sept. 86
14. John F. SOWA, Conceptual Structures: Information Processing in Mind and Machine. Addison-Wesley, London, 1984
15. M. Willems, Generalization of conceptual graphs. In Proceedings of the 6th Annual Workshop on Conceptual Graphs, pp. 27-37, July 11-13, 1991.
16. D.A. Waterman and F. Hayes-Roth, editors. Pattern-Directed Inference Systems.. Academic-Press, New York, 1978
17. J. Bouaud and P. Zweigenbaum, A Reconstruction of Conceptual Graphs on Top of a Production System. In Proceedings of the 7th Annual Workshop on Conceptual Graphs, July 8-10, 1992.