A KADS refinement for explanatory knowledge extraction and modelling

Philippe MARTIN

INRIA Sophia Antipolis - Project ACACIA
2004, Route des Lucioles - B.P. 93 - 06902 Sophia-Antipolis Cedex - France

ABSTRACT

In order to help the knowledge engineer to extract and organize explanatory and cooperation knowledge when building an Explainable Knowledge Based System, we propose : 1) an extension of the KADS Conceptual model;
2) an explanatory knowledge extraction method which relies on KADS interpretation models.

1 Introduction

This article proposes somes *methodological* specifications on extraction and modelling of knowledge for a Knowledge Based System (KBS) which should provide user suited explanations, i.e. at a sufficient abstraction level and related to the user goals and habits.

In KADS, the analysis phase leads to a «Conceptual Model» consisting of a *model of (problem solving) expertise* and a *model of cooperation*. The first one models the necessary problem solving knowledge at a «knowledge level» [10]. Then in the design phase, these models are implemented but their high level structure are preserved. Thus, the KBS explanatory tasks may explain at a high level the KBS results or reasonings by retrieving original abstract knowledge from the used code.
In the ***first part*** of this article we will focus on the structure and types of knowledge that the Conceptual Model should have, if the KBS has to provide «good» explanations. In this way we guide the extraction and modelling of knowledge needed for explanations.
The ***second part*** supplements this help in acquisition, by giving a questionnaire which enables to collect much information for instantiating an interpretation model. Both problem solving and explanatory information are collected with this method.

2 An Extended Conceptual Model

2.1 Weakness Of The KADS Standard Analysis Of Cooperation

In KADS-I, the *model of (problem solving) expertise* is modelled with a four level language : KCML (KADS Conceptual Modelling language). In KADS-II, a more formal language is used. But the *model of cooperation* is not structured as a model of expertise since it only lists and roughly defines the *transfer tasks*, i.e. tasks managing exchanges of information between the system and the user. In order to specify the interaction functions used by these transfer tasks, in [3] de Greef and Breuker propose to build another model, the «communication model» which would have two great parts : a *presentation* manager that takes care of interpreting or generating expressions, taking into account the intended effects, and a *dialogue* manager that controls topics and turn taking (possibly taking into account the fact that a same communication intention may concern multiple interactions with the user).
Despite the complexity of these tasks and the amount of knowledge they could imply, de Greef and Breuker *do not propose* to structure the communication model like a model of expertise. Moreover this model could directly include details of the interface with the user and then would be implementation-dependent like design phase models.

We think that if the communication expertise is sufficient, the model of communication should be structured like a model of expertise, for example with KCML. Then, during the design phase, this model can be refined in order to include the design constraints. Hence we suggest to include to the Conceptual model, a «model of communication expertise» which we abbreviate by **Communication-ME**.

KADS proposes to specify the alternation between system problem solving tasks and transfer tasks, by an ***algorithm***. This algorithm is then inserted to the model of (problem solving) expertise at its task (and strategy) levels. This method is simple when the expected system-user cooperation is rigid and hence may be statically defined, but in *other cases* we think that an *additional model* of expertise is needed for modelling the concepts, rules, tasks and strategies of cooperation management.

- Actually, a «real cooperation» management, e.g. an interactive work distribution between the system and the user, implies ***complex*** reasonings depending on numerous ***factors*** : knowledge and capacity of the user, his goals, the various tasks the system should accomplish or should lead the user to accomplish in order to get data or to teach him something. Furthermore, since the user progresses and since the priorities between the different goals may change, the cooperation should be dynamically managed according to these various parameters or factors.
  If cooperation is dynamically managed, it means that tasks are negotiated and planned with the user as it is already done in some KBS or tutoring systems.
- The cooperation knowledge constitutes an expertise distinct from the problem solving expertise, which therefore needs to be ***modelled in a distinct model of expertise***. As a matter of fact, it would not be much easy or modular to mix in a single model of expertise, knowledge with such different roles : problem solving vs cooperation. We call this new model, which may be structured with KCML, **Control-ME** since it calls and controls both problem solving tasks and communication tasks, in order to reach a fixed cooperation goal (see figure 2).

Cooperation is an important aspect of explanations since it controls at a high level the system-user interactions. For example, the system may introduce new topics in the dialogue. To sum up, for better specifying expertises related to explanations, i.e. the communication (and the cooperation) expertises, ***we propose to build a*** **Control-ME** ***and a*** **Communication-ME,** ***instead of the (so called) «model of cooperation»***. Similarly, we will note **Solving-ME** the «model of (problem solving) expertise».

Our extension is intended to enable the knowledge engineer to model other expertise than the problem solving one. Therefore :
1) this extension is not useful if the intended communication or cooperation with the user can be easily defined. If a good communication is expected but not a real cooperation management, the Control-ME can simply be not built;
2) this extension is not dependent on the language used to model the expertise, and hence can be applied on KADS-II Conceptual Model (moreover cooperation modelling in KADS-II is still the same as in KADS-I [3]).
3) our specifications are ***at the same abstraction level as KADS***, hence they are *methodological* and independent of any implementation.

Before elaborating our two new models, the next section shows some relations which they have with each other, and the data on which their tasks work on.

2.2 A New Architecture For The Conceptual Model

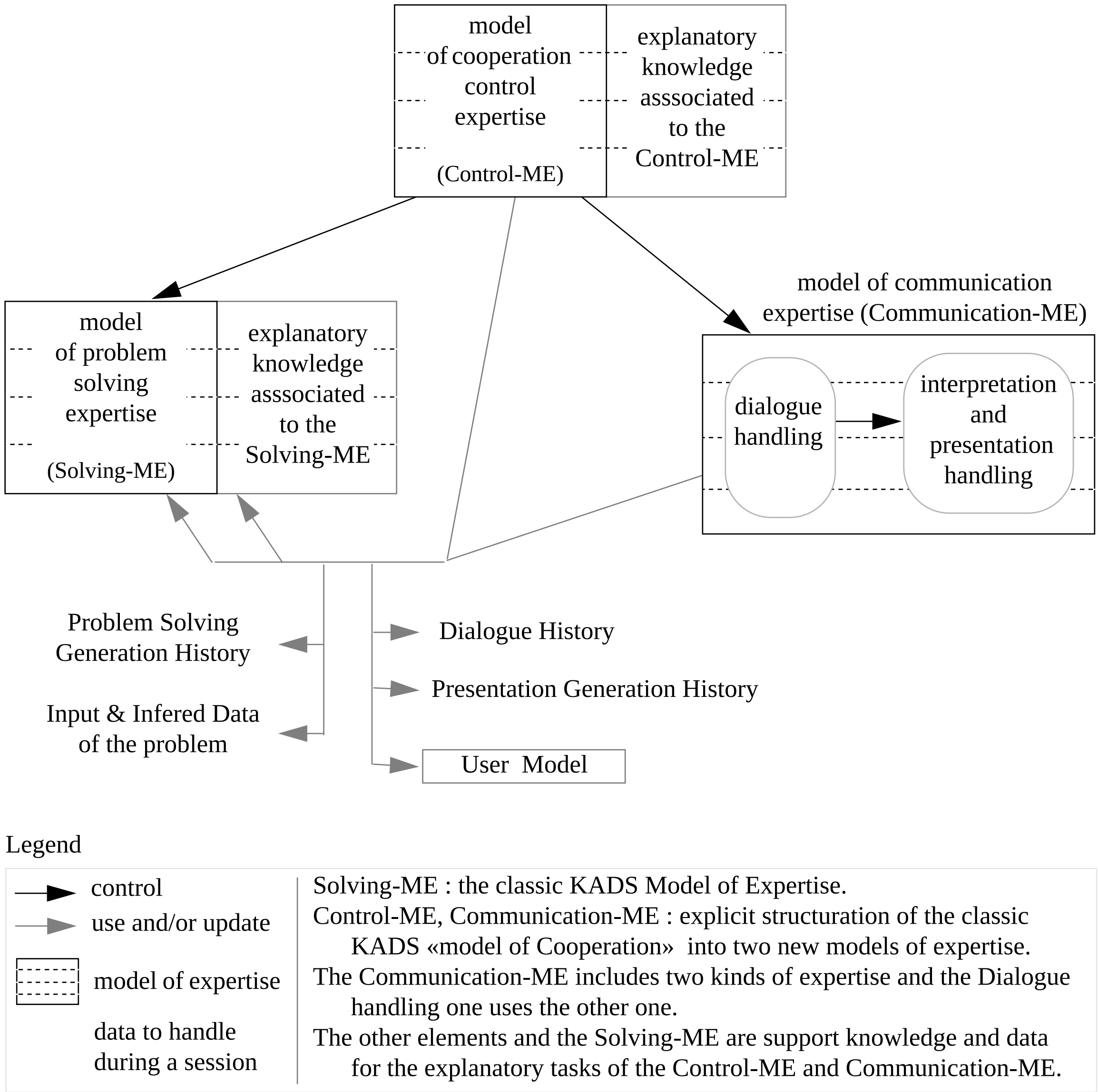

Figure 1: Architecture of submodels of expertise for the Conceptual model

This figure 1 ***shows*** that the Control-ME triggers the solving and communication tasks for managing the cooperation with the user, and ***lists*** data on which tasks of the Control-ME and of the Communication-ME may work. This list is not exhaustive but all its elements are ***necessary*** ([8][9][11]) for generating high level and user suited explanations, and each one should be *exploited* as much as possible. Hence mechanisms for this *exploitation* must be ***acquired or defined*** according to the explanation needs. (But during the design phase, the run time dedicated to exploit the support knowledge must be minimized and hence all its resources may not be exploited). Here are these ***«explanation (and cooperation) support» knowledge and data***.

- ***Solving-ME*** : we saw that KADS enables to link «***Input & Infered Data***» and elements of *traces of reasonings* with the high level elements of the Solving-ME - concept, relation, meta-classes, goal, task, etc. In [1], the authors show how to exploit each element of a «model of expertise» for interpreting user requests and answer them adequately. Sprenger [12] follows up this work and gives several algorithms for exploiting the various levels of the model *depending on* the aimed explanatory goal.
- ***Problem Solving Generation History*** : the abstracted and structured *problem solving traces* needed for explanation and cooperation reasonings. It includes : tasks, goals, methods, choices done by the system with their justifications and consequences, etc.
- ***Explanatory knowledge associated to the Solving-ME*** : for good explanations, descriptions of the Solving-ME need to be commented or completed by meta-knowledge, which may be structured as a model of expertise. With KCML, its *domain level* essentially includes the deep and/or explanatory knowledge *associated* with the domain level knowledge of the Solving-ME but which is not necessary for problem solving, e.g. causal justificative networks, definitions, explanatory schemas, figures, etc. This *domain level* may also include «explanatory interpretations or schemas» of elements of other levels of the Solving-ME. Thus, inferences, tasks and strategies may be explained in the expertise domain language. The *other levels* of this model of expertise are by definition independent of the expertise domain and hence may include explanatory schemas [4] on the interpretation models used in the Solving-ME.
  These descriptions are also valuable for the explanatory knowledge associated to the Control-ME if it is collected.
- The «**User Model**» (UM) : it may be developped by the designer and/or updated after each interaction or session with the user. A UM consists of (approximate) knowledge on the user : goals, centers of interest, habits, methods, types of questions, knowledge on the domain, types of difficulties and/or aptitudes for understanding or learning, preferences, etc. (a part of this knowledge may be acquired automatically) [8].
- The «**Dialogue History**» : it is a structured record of the system-user exchanges, e.g. data, requests, information or explanations. It enables the *interpretation* of implicit elements in requests. Furthermore, the *cooperation* and *dialogue* handling tasks rely on this history for determining subjects being discussed or to be discussed.
- The **«Presentation Generation History»** : it associates given explanations to goals sought and choices made for their generation. Such an history is essential for handling a dialogue or answering follow-up questions (i.e. questions on previous explanations) [9][1]. A similar structured trace could perhaps be useful for dialogue handling tasks.

KADS does not forbid to build several «models of (problem solving) expertise» if different kinds of problem solving expertise are needed in an application. If he wishes such a modularity, the knowledge engineer can easily adapt our architecture by building as many «Solving-ME and explanatory knowledge associated to it» as he wants. The other models of expertise have not to be modified since their tasks exploit the knowledge of a Solving-ME as data (and the tasks of the Control-ME will manage the various problem solving tasks according to the type of the given problems).
The *modularity* of this architecture and the KADS language for modelling an expertise, enable to extract and to structure adequately the various needed expertises. Moreover, most techniques which apply to the Solving-ME, are also usable for the other models of

expertise, for example : a) techniques exploiting a model of expertise for explanations ([1][12]); b) extraction, design and implementation techniques.
The acquisition of cooperation and communication expertise is therefore guided.

2.3 The model of cooperation Control expertise

For large cooperative systems, the Control-ME should include concepts, rules, methods, tasks and strategies for *distributing tasks*, for *negotiating this distribution* with the user, and for *controlling at a high level other interactions* with the user such as information exchanges and explanations. The Control-ME ***uses and directs*** tasks of the Communication-ME by the following ways : 1) it ***determines*** the «*messages*» to communicate to the user by ***setting*** more or less *subjects or concepts* of the discourse; 2) it ***determines*** «when to modify» or «when is modified» the *main goal of a current transaction* with the user; for Cawsey [1], these two tasks must be done by a module separate from the dialogue manager module; therefore we naturally assigned them to the Control-ME.

The *actual* explanation generation is hence mostly handled by the Communication-ME. This is due to the actual working ways and possibilities of what the Communication-ME has to model : present techniques in dialogue managing and discourse generation (these techniques look alike and work on discourse goals).

2.4 The model of Communication expertise

The Communication-ME includes a Dialogue expertise and an Interpretation/Presentation expertise. This last one models the *complex and implementation-independent* functions of the two main classes of transfer tasks : data reception and transmission.

- Interpretation consists of translating data (including request) expressed in a «language» understandable by the user (e.g. menus, keywords, request language, natural language), into a data structure the format of which is recognized by the system. Hence the ***domain and inference levels*** of the Communication-ME must include the different *concepts and methods* related to : 1) the user «language» and the system «language» (the domain level must include the lexicon, the syntax, the semantics and the pragmatics of each language); 2) the translation from a language to another and the use of the «explanation support knowledge» for this translation;
  At the task level, the knowledge engineer defines the sequencing of these methods - more precisely, of the KS (Knowledge Sources) which model the methods - according to the expected flexibility in the transformation, the interactivity with the user, etc.
- Presentation consists of providing information on some objects of the system, in a language understandable by the user. Hence this language must be defined and knowledge for discourse generation with this language must be added.
  In [7] we gave general inference structures for interpretation and presentation. For presentation, we also proposed a more detailed inference structure and we advocate modelling of *discourse generation* «operators» ([9][11]) at the domain level or as methods of KS. Then we discussed various possible controls on these operators.

Dialogue expertise is less easy to define than presentation expertise since it is even more dependent on the application : expected type of cooperation, chosen type of dialogue, etc. Moreover, the present state of the art on cooperative dialogues does not enable us to give inference structures such as for presentation. But some discourse generation control techniques may also apply for dialogues [1] and we show in [7] how to model them within KADS. For dialogue, the Communication-ME should include the following elements (more or less developed in accordance with explanation needs) :

- ***at the domain*** level : a dialogue model (conventions, design and cooperation rules) and all the other concepts and relations necessary to the inference and task levels;
- ***at the inference, task, and strategy*** levels : all the mechanisms which will manage topics, turn taking, User Model exploitation and goal application (the goal comes from the Control-ME or from a user request interpretation) which include : 1) *spontaneous explanations* (given to anticipate foreseeable user questions); 2) *adaptation* to new user aims or knowledge; 3) *updating and exploiting* the dialogue context (earlier subjects, requests, explanations, etc.).

We see that dialogue tasks use presentation tasks by giving them topics to explain. Hence in order to be useful for dialogues, presentation tasks should support interruptions and be able to continue or improve an explanation [1].

During the analysis phase, modelling is more or less accurate. For example, interruption handling may be modelled by a simple knowledge source, the methods of which can be specified later in the design phase. Therefore during the analysis phase, the Communication-ME is quite independent from the final user interface.

To sum up, this first section showed which elements are necessary for explanations, the relations that link them and where they can be modelled within KADS. In the next section, we supplement this help by giving a method for extracting complete explanatory knowledge, especially explanation support knowledge and data.

3 Knowledge To Extract For Instantiating KADS Interpretation Models

Extracting and modelling knowledge for a Model of Expertise will also generally provide constraints and knowledge for other models of expertise. The method we propose is not oriented for eliciting a particular expertise : it relies on interpretation models used by the knowledge engineer in order to be guided when extracting and modelling the expert knowledge, and hence ***can be used on any interpretation model***. For each element of this model, our method lists via a set of questions, *the knowledge needed to instantiate it and also the explanatory meta-knowledge which comments this knowledge*. Actually, instead of questions, any elicitation technique may be used : interviews, observations, Wizard of Oz, etc. Furthermore, the pointed knowledge may be extracted ***in many steps***, for example by successive refinements.
Even when no appropriate interpretation model can be found for modelling expertise, this method gives good ideas on what shoud be obtained.

The proposed questions are very general. Hence they must be adapted by the knowledge engineer to the ***domain*** of the expertise, and some will not be appropriate. Furhermore, they include common terms in order to help the knowledge engineer to adapt his language to the expert's one. Then, words such as «concept», «relation» or «role» have common meanings. But remarks on modelling are expressed in KADS terms.

FOR each interpretation model IM (in order to model an actual task, the chosen IM must be filled and adjusted; the following questions are intended for that and also for collecting explanatory meta-knowledge)

ask goals, interest and principles of the actual task or strategy
Instantiate or adjust each KS (Knowledge Source) and meta-class of IM (see «A:», «B:» and «C:» below; thus the inference level of ME is updated, its domain level

filled, and the «explanatory knowledge associated to it» supplemented)
ask about when to use the task inferences and their sequencing (update ME's task level)
ask about when to use this task and in which context (update ME's strategy level)
(the «A:» questions are easily adaptable for asking about strategy or task control).

A: Instantiate or adjust a KS (i.e. an elementary inference or *function*)
Taking into account the context of this KS (e.g. its position in IM inference structure)
ask whether there actually exists such a function : if yes, what is it used for, what are its principles, when is it needed and when are other functions needed ?
if not, update this KS or add a new KS to IM
instantiate or adjust the KS input and output meta-classes (see «B:»)
what are the actual methods that realize this function ? (the answers will enable to model the «inference methods» or «domain view» of the KS)
what are their common features, what are the criteria for choosing a method or another ?
for each method, ask about its description and its triggering context, for example :
- which domain relation (functional, spatial, ...) does it use and could it possibly uses ?
- what are its steps, its submethods; is it used in other KS ?
- what are its triggering context : events, links with concepts, etc ?
- what are the criteria for assessing its results ?
- what is its limits, can it be improved, how to recover from a failure ?
- for which other goals may it be used, may its principles be generalized ?
with this collected information, instantiate the KS and supplement the explanatory knowledge associated to ME.; other levels of ME may also be updated.

B: instantiate or adjust a meta-class (its name underlines the role of the concepts it refers to)
Taking into account the context of this meta-class (e.g. its position in the IM inference structure)
ask whether concepts or objects have a similar role before, during or after the solving process : if yes, ask their common features, else update the meta-class or add a new one to IM
ask specifications on the actual role of the refered concepts or objects
for each concept or object, ask about its description and its triggering context (hence of course, some concepts will already be described), for example :
- with which concepts or concept classes is it related ?
- which other roles does it play during the process described by IM and in other ones ?
- how is it handled by this process or by others ?
- what are its properties and how to update or preserve them during the process ?
- what are its ranges and default values ?
with this collected information, instantiate this meta-class, supplement the domain level of ME and supplement the explanatory knowledge associated to ME.

C: extract explanatory meta-knowledge on an information (during or after «A:» and «B:»)
For each information given by the expert (concept, criterion, role, task, goal, etc.) :
- which kinds of user know this information and how do they use it ?
- is it important in the expertise domain or during the solving process ?
- from which viewpoints may it be seen ?
- in order to explain it, which knowledge must be used, in which order, in which context and for which kinds of user ?
- are there usual schemas for explaining it, and how to choose among them ?
- for which explanation types, will this information be useful and in which context ?
- if the information is a choice, is it good or just acceptable ? What alternative choices ?

For **KS**, we found [6] additional some question types which perhaps could not be easily infered from the generic questions that we proposed. Here is an example.

- `generalize(set of instances -> concept)` (find a concept which covers features of instances) :
  - is the concept already known, always known ?
  - what are the criteria for generalization, what must be done if more than one concept is a good candidate for the generalization ?
  - which object or object features are more important for generalisation than the others ?
  - what are the criteria for handling exceptions in values or object features ?
  - what must be done if some values or object features cannot be acquired ?

Information collected with this method will enable the explanatory tasks to give several viewpoints [5] on each element of the Solving-ME : *categorical* (how a concept fits a hierarchy), *functional* (role of an object in a process), *structural* (parts of object, steps of process), *modulatory* (how an object or a process affects another object or process), etc.

4 Conclusion

We listed «explanation support knowledge and data» necessary for really cooperative or user friendly systems, and showed how to extract and structure them using KADS techniques and framework. Then we gave a typology of questions which seems more accurate for knowledge acquisition than classic ones.

5 Acknowledgments

We are grateful to Olivier Corby, Rose Dieng, Alain Giboin and Sofiane Labidi for their comments on earlier versions of this paper.

6 References

-> AAAI-W 91: *Proceedings of the AAAI'91 Workshop on Comparative Analysis of Explanation Planning Architectures*, Anaheim, California, July 1991.

[1] A. Baumewerd-Ahlmann, P. Jaschek, J. Kalinski and H. Lehmkuhl, *Embedding Explanations into Model-Based Knowledge Engineering - Improved Decision Support in Environmental Impact Assessment*, Proc. of KMET'91 IOS Press, Sophia-Antipolis, France, April 1991.

[2] A. Cawsey, *Generating Interactive Explanations*, AAAI-W 91, pp. 6-14.

[3] P. de Greef and J. Breuker, *Analysing system-user cooperation in KADS*, Knowledge Acquisition (1992) 4, pp 89-108.

[4] B. Lemaire and B. Safar, *An architecture for representing explanatory reasoning*, Proc. of KMET'91, Sophia Antipolis, France, April 1991.

[5] J. C. Lester, B. W. Porter, *An Architecture for Planning Multi-Paragraph Pedagogical Explanation.* AAAI-W 91, pp. 27-41.

[6] Ph. Martin, *La méthodologie d'acquisition des connaissances KADS et les explications*, to appear in INRIA research report, rapport de stage de D.E.A. informatique, INRIA Sophia Antipolis, 1993.

[7] Ph. Martin, *Adaptation de KADS pour la construction de Systèmes à Base de Connaissances explicatifs*, JAC-93, Saint-Raphael, March-April 1993.

[8] M. T. Maybury, *Customs Explanations: Exploiting User Models to Plan Multisentential Text*, Proc. of the Second Int. Workshop on User Models, University of Honolulu, Hawaii, march 1990.

[9] J. D. Moore and C. L. Paris, *The EES Explanation facility: its tasks and its architecture*, AAAI -W 91, pp. 65-79.

[10] A. Newell, *The knowledge level*, Artificial intelligence, Vol 19, N° 2, 1982, pp. 87-127.

[11] D. D. Suthers, *Task-Appropriate Hybrid Architecture for Explanation*, AAAI-W 91 pp. 80-94.

[12] M. Sprenger, *Explanations Strategies for KADS-based Expert Systems*, Technical Report 10, Universitat Bielefeld, GMD, Sankt Augustin, Germany, July 1992.