Building a decision support system with a knowledge modeling tool

Martin Molina

Department of Artificial Intelligence
Universidad Politécnica de Madrid
Campus de Montegancedo S/N
28660 Boadilla del Monte, Madrid, Spain
mmolina@fi.upm.es

ABSTRACT. *Knowledge modeling tools are software tools that follow a modeling approach to help developers in building a knowledge-based system. The purpose of this article is to show the advantages of using this type of tools in the development of complex knowledge-based decision support systems. In order to do so, the article describes the development of a system called SAIDA in the domain of hydrology with the help of the KSM modeling tool. SAIDA operates on real-time receiving data recorded by sensors (rainfall, water levels, flows, etc.). It follows a multi-agent architecture to interpret the data, predict the future behavior and recommend control actions. The system includes an advanced knowledge based architecture with multiple symbolic representation. KSM was especially useful to design and implement the complex knowledge based architecture in an efficient way.*

KEYWORDS: *decision support systems, knowledge modeling tool, knowledge acquisition tool*

This version corresponds to a preprint

of the actual paper to be published in the

Journal of Decision Systems, Lavoisier, 2006.

1. Introduction

Knowledge representation techniques from the field of artificial intelligence have been widely used in the development of decision support systems (DSS). The design of such a systems in complex domains usually includes the integration of heterogeneous solutions that need to be adequately structured to produce efficient architectures. In the field of knowledge engineering, several proposals based on the idea of *knowledge modeling* have been done to help in the development of such a complex systems. For instance, knowledge engineering methodologies such as CommonKADS (Screiber et al., 00) follow the modeling approach in the *analysis* phase to produce a knowledge model that describes in detail the knowledge of the system using a uniform perspective based on intuitive description entities. In addition to that, the final *design* and *implementation* can be also facilitated with the help of a new generation of tools, for instance the KSM tool (Molina, 93), that provide a set of pre-programmed building blocks to be assembled following the uniform perspective provided by the knowledge modeling approach.

The purpose of this article is to show the advantages of using knowledge modeling tools such as KSM in the development of a DSS. We illustrate this with a case that belongs to a type of DSS whose goal is to help users in the surveillance and control of a complex dynamic system (e.g., a road traffic network, a basin with river channels and reservoirs, a public transport network, etc.). In this type of problem, the DSS must include an efficient model of the dynamic system to automatically provide on real time answers about future hypotheses of behavior. The DSS also must be able of providing explanations about its conclusions given that the operator takes the final responsibility of decisions. In addition to that, it is also interesting to design a solution with certain generality to be reusable in the development of different specific realizations.

The DSS described in the article is a knowledge-based system, called SAIDA, for the management of emergencies produced by floods. The system was developed within the SAIH National Programme (Spanish acronym for Automatic System Information in Hydrology). The SAIH Programme was developed in Spain with the goal of installing sensor devices and telecommunications networks in the main river basins to get on real time in a control center the information on rainfall, water levels and flows in river channels. One of the main tasks in this type of control centers is to help to react in the presence emergency situations as a consequence of river floods. SAIDA was developed to help operators in these situations.

According to this, the article shows first the functional model of SAIDA describing the main tasks (evaluation, prediction and recommendation) that were distributed according to a multiagent architecture. Then, the article shows an overview of the KSM tool. Then, the article shows how SAIDA was developed using the KSM tool. Finally, a discussion about the advantages of using a tool like KSM in the development of the DSS is presented.

2. The knowledge model of the decision support system

SAIDA is a decision support system that was designed to help operators in hydrologic control centers for the special case of emergencies produced by floods. SAIDA was designed and developed with the help of KSM. This section presents an overview of the knowledge model of SAIDA Then, the next sections, will provide details about the implementation of SAIDA using KSM.

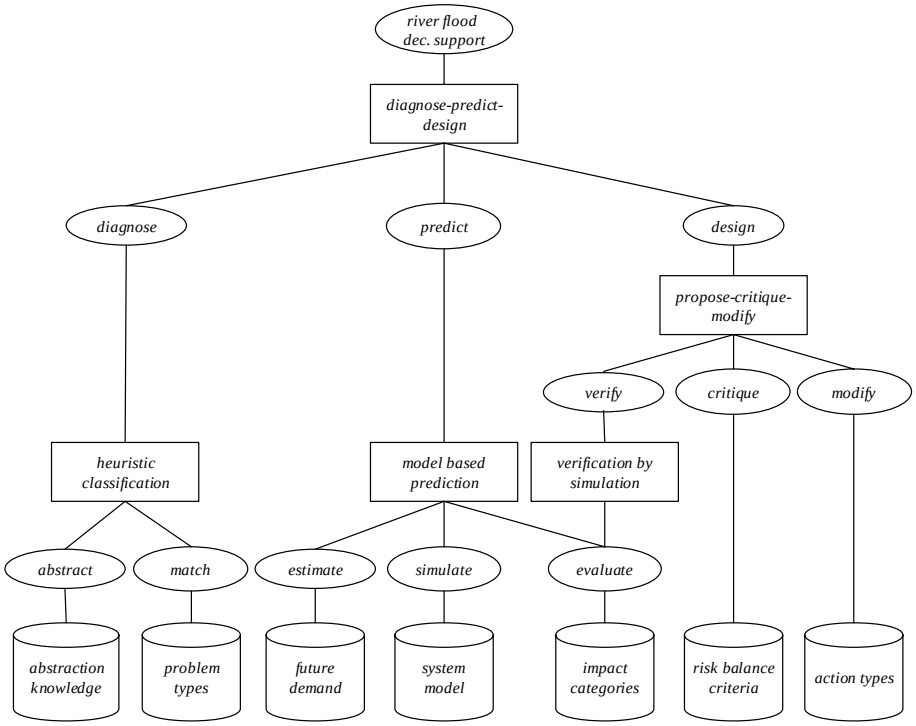

Figure 1. *The task-method-domain structure of the SAIDA system. Legend: circle (task), square (method), cylinder (type of knowledge base).*

To describe the knowledge of SAIDA we use here similar types of description entities that are present in current methodologies for knowledge engineering, for instance, CommonKads (Schreiber et al., 00). According to this view, a *task* is an abstract description that identifies a goal to be achieved (for instance, mineral classification or the design of the machinery of an elevator). On the other hand, *problem-solving methods* (or *methods* in short) indicate how a task is achieved, by describing the different reasoning steps by which its inputs are transformed into outputs. Simple tasks can be performed directly using declarative knowledge. This

requires an ontological definition of such a declarative knowledge that is viewed as a set of *domain models* in form of types of knowledge bases that support primary tasks. This type of description based on tasks and methods was originally present in several proposals from different authors such as the *generic task* (Chandrasekaran 83, 86), the KADS model (Wielinga et al. 92), the model of components of expertise (Steels 90), the *role limiting method* (McDermott 88).

Following this approach, a model can be described with one or several *top-level tasks* that identify the set of main goals to be achieved by the application. These tasks requires methods that decompose them into subtasks. These subtasks may again be decomposed by a method and so on, developing a task-method-domain hierarchy, whose leaves are given by basic tasks that use simple knowledge bases. Thus, a knowledge model can be viewed as a collection of types of knowledge bases (each one with its own symbolic representation) together with a hierarchically structured set of reasoning strategies that make use of such knowledge bases.

Figure 1 describes the knowledge model that we designed for SAIDA (Molina, Blasco, 03). SAIDA operates in a control center where hydrologic data (e.g., rainfall, flows, water levels, etc.) from different locations in the basin are received periodically (for instance, every half an hour). SAIDA receives this information as input and performs three main tasks: evaluation, prediction and recommendation. The goal of the *evaluation task* is to identify potential emergency situations by interpreting sensor data about the current state of the river basin, based on certain patterns of scenarios about rain, water levels, flows and reservoir states. This corresponds to a typical classification task that selects a category (a type of problem) within a prefixed set, based on a set of observations (sensor data). In this case, this task can be solved by an adaptation of the *heuristic classification* method (Clancey, 85) with two steps: (1) *abstract* to abstract data from sensors, and (2) *match* to find patterns of problems that match the current information from sensors.

The goal of the *prediction task* is to predict the future behavior and estimate potential damages. This task can be carried out by a method that performs the following steps: (1) *estimate future rain* to generate hypotheses of future rain for the next H hours (e.g. H = 8 hours), based on a heuristic model about the river basin and the global weather forecast received as input, (2) *simulate the river behavior* that uses a model of the river basin based on causal relations, and (3) *estimate potential damages* to estimate the impact of the flows in terms of potential damages by using relations between flows and qualitative ranges of severity for each particular critical location.

The goal of the *recommendation task* is to suggest possible control actions as an answer to the detected problem. This distinguishes between two possibilities: (a) *hydraulic actions*, that establish discharge policies at the dams to avoid undesirable impacts and (b) *defensive actions*, such as population alert, evacuation procedures, etc. involving different organizations like traffic police, health services, fire brigades, army, etc. The first case can be performed by a method that explores a

search space of potential hydraulic actions using a heuristic approach. The basic idea is that the method evaluates the current situation and, based on empirical knowledge, proposes a set of hydraulic actions that potentially can solve the problem. Then, these actions are tested by simulation and, if the result of the test is not satisfactory, empirical knowledge is used again to modify the hydraulic actions. These steps are performed in a loop until a satisfactory set of control actions is found. In the artificial intelligence literature, this method receives the name of *propose-critique-modify* (Brown, Chandrasekaran, 83; Marcus, McDermott, 89). The set of defensive actions is found by using a classification method supported by a model that relates types of problems with types of defensive actions (for the shake of simplicity, this method is not included in the figure corresponding to the complete model).

Types of agents	**Types of knowledge bases**
Problem detection agent	Abstraction
	Problem types
	Impact categories
	Risk balance criteria
	Action types: agent relations
Reservoir management agent	Abstraction
	Problem types
	Impact categories
	Risk balance criteria
	Action types: discharge strategies
Hydraulic agent	Abstraction
	Future demand
	System model
Protection agent	Action types: transport network
	Action types: population
	Action types: constructions

Figure 2: *Knowledge bases for each type of agent.*

We followed the concept of multi-agent system to complement the previous knowledge model. We identified four types of agents: (1) *hydraulic agents* to give answers about the behavior of the physical process, (2) *problem detection agents*, to evaluate the flood risk in a particular geographical area, (3) *reservoir management agents*, with criteria for exploitation strategy for each reservoir, and (4) *civil protection agents*, responsible to provide with resources of different types according to the demands of the problem detection agents. Figure 2 shows how we distributed the different knowledge bases between types of agents.

3. The knowledge modeling tool

This section provides an overview of the main characteristics of the KSM tool. More details about this tool can be found at (Molina, 93; Cuena, Molina, 00; Molina, Cuena, 04). KSM (Knowledge Structure Manager) is a knowledge modeling tool that includes and extends the paradigm of task-method-domain followed by different knowledge engineering methodologies. In real applications, the experience shows that, sometimes, too large descriptions can be designed by using only the task-method-domain formulation. It may produce problems of understanding and maintenance together with problems of efficiency in the final software implementations. Thus although the conceptual description based on task-methods-domains is adequate for the *analysis* process, it needs to be complemented and re-organized using additional modeling concepts for the final *design*. Here, the design metaphor of agent society is useful to provide global views. However, within each particular agent it is still necessary to provide additional modeling components to complete the design. For this purpose, KSM uses the entity called *knowledge area* that follows the intuition of a body of knowledge that explains a certain problem solving competence.

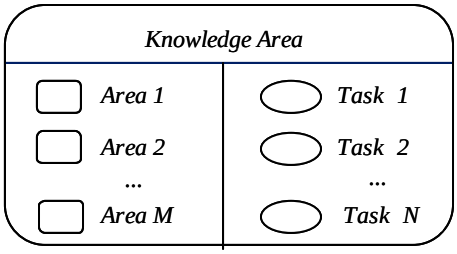

Figure 3: *The structure of a knowledge area*

A knowledge area (figure 3) is described with two parts: (1) its knowledge, represented as a set of component sub-areas of knowledge, and (2) its functionality, represented by a set of tasks (and their corresponding methods). The first part decomposes the knowledge area into simpler subareas, developing a hierarchy at different degrees of detail. The second part associates tasks to knowledge areas showing their functional capabilities. The knowledge area concept is useful to produce a more synthetic view of the knowledge model given that it integrates a set of tasks (together with the corresponding problem-solving methods) in a conceptual entity of higher level.

The whole knowledge model is viewed as with a top-level area that is divided into other more detailed sub-areas (using part-of relation) that, in their turn, are divided into other simpler areas and so on, developing the whole hierarchy. Knowledge areas can be also defined at *generic* and at *specific* level. Generic areas

mean classes of bodies of knowledge that allow to formulate a model. Then, a particular specific model is viewed as a collection of instances of such classes that can share by inheritance different properties of the classes such as relations with other areas, problem-solving methods, etc.

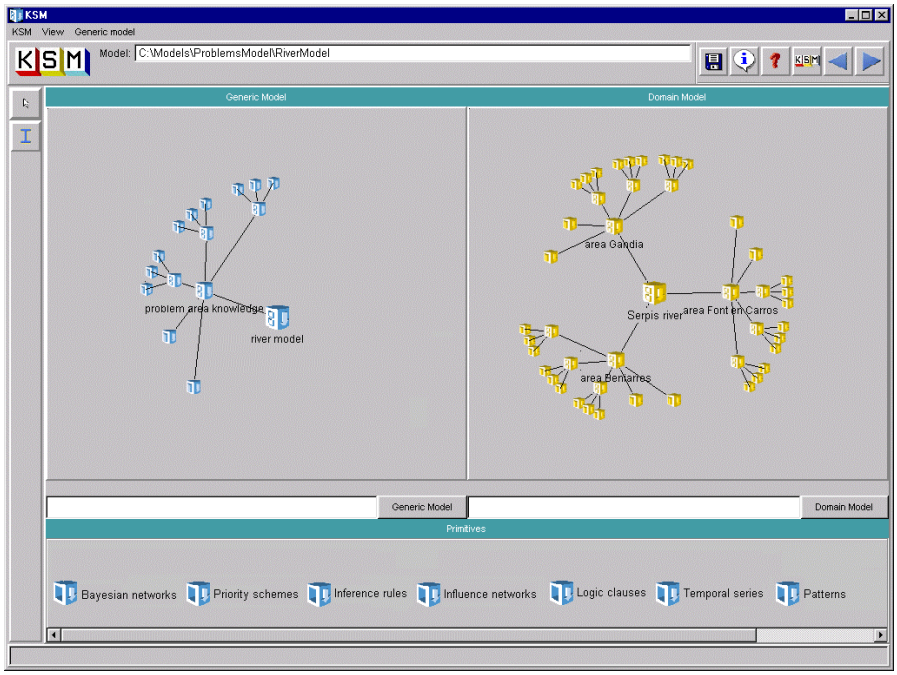

Figure 4: *Main screen of the KSM tool.*

One of the basic assumptions followed by the KSM approach to develop a knowledge model is that, instead of using a uniform symbolic knowledge representation for the whole model (e.g., logic or rules) that can be useful for the analysis and formalization phases but could be artificial and inefficient for the development of the final application, the developer will use for each case the most appropriate symbolic representation in order to produce both an efficient and a comprehensible model. According to this, KSM provides a library of reusable software components, called *primitives of representation* (Molina et al. 99), that offer the required freedom to the developer to select the most convenient representation for each case (rules, frames, constraints, belief networks, etc.). The primitives of representation are taken from an library of primitives provided by KSM. This library is open, i.e., new primitives can be included as a result of the development of new software components. Each primitive of representation is considered as a reusable pre-programmed software component that implements a generic technique for solving certain classes of problems.

KSM conceives the final application as a modular architecture made of a structured collection of knowledge areas. At the implementation level, each elementary area is supported by a reusable software component programmed with an appropriate language and a particular technique (knowledge-based or conventional). Using KSM, a developer can duplicate, adapt and assemble the different software components following a high level knowledge model which offers a global view of the architecture. Figure 4 shows the main screen presented by the KSM tool.

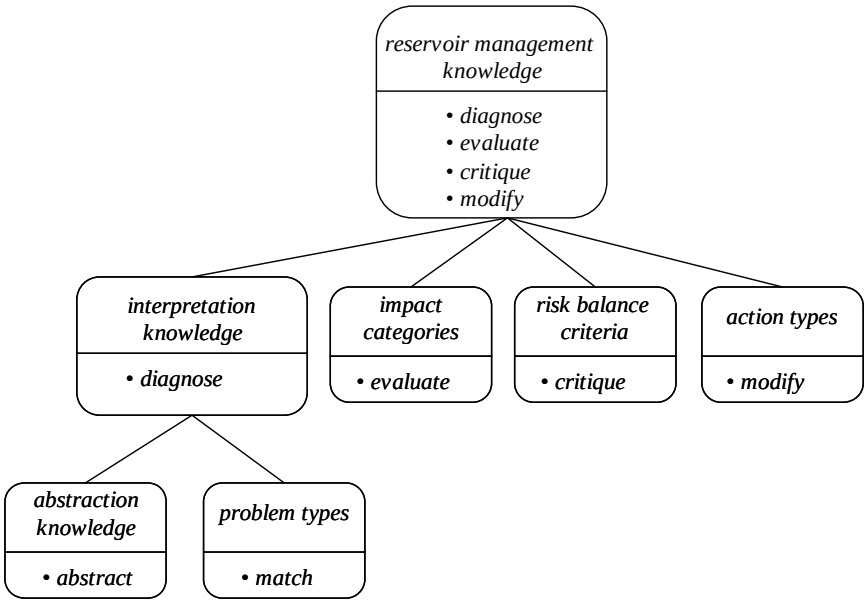

Figure 5: *Structure of knowledge areas for the reservoir management agent (partial view).*

4. Development of SAIDA with the KSM tool

In order to construct the SAIDA system with the KSM tool we formulated four complete knowledge models, one for each type of agent (reservoir management agent, problem detection agent, hydraulic agent and protection agent) as a structured collection of classes of knowledge areas. The task-method-domain structure (figure 2) was grouped into modules following the format of the knowledge areas in such a way that the different tasks and methods were encapsulated into knowledge areas within each type agent.

Thus, for example, figure 5 shows a partial view of the structure of knowledge areas for the case of the reservoir management agent. The figure shows a top-level area, *reservoir management knowledge*, that represents the whole knowledge of the reservoir management agent. This area includes a set of tasks (*diagnose*, *evaluate*, etc.) that show the global functionality provided by the agent. The area is decomposed into simpler areas (part-of relation), for example, *interpretation knowledge*, *impact categories*, etc. Each subarea includes other simpler tasks. Normally, as it is shown in this example, the names of primary areas at the bottom level correspond to the types of knowledge bases identified by the task-method-domain model of figure 2. Similarly, the tasks of primary areas correspond to primary tasks in the task-method-domain model of figure 2.

Intermediate and top-level areas include intermediate tasks (e.g. *diagnose*) that are performed by problem-solving methods. In KSM methods are formulated using a particular language called *Link* (Molina et al., 98b). This language allows developers to specify the control-knowledge of the problem-solving methods. For instance, figure 6 shows the particular adaptation of the heuristic classification method in this problem to perform the diagnose task of the reservoir management agent. In the example, data flow section defines how tasks are connected (where tasks are associated to knowledge areas). The control flow section uses rules to establish the execution order of the tasks. In this case there is only one rule that executes two tasks in sequence.

```
METHOD heuristic-classification
   INPUT observables
   OUTPUT problems

DATA FLOW
  (abstraction knowledge) abstract
      INPUT observables
      OUTPUT abstractions
  (problem types) match
      INPUT observables, abstractions
      OUTPUT problems

CONTROL FLOW
   START
   -> (abstraction knowledge) abstract,
      (problem types) match,
      END.
```

Figure 6: *Example of problem-solving method formulated using the Link language.*

The generic model for each agent also includes a set of common concepts that are shared by different knowledge areas. In KSM these vocabularies are formulated using a particular language called *Concel* that uses a concept-attribute-facet representation together with an organization in classes-subclasses and instances. In the model of the reservoir management agent there is a vocabulary that includes the set of general concepts, for example, sensor devices such as pluviometers, flow

sensors, water level sensors, or general concepts to define the structure of the basin such as rainfall area, section of river, reservoir, etc.

To be supported by computational resources, this conceptual structure for each agent was refined by selecting appropriate symbolic representations and inference procedures. For this purpose, we used the concept of *primitive of representation* provided by KSM. Each primitive defines a particular representation language together with knowledge acquisition facilities and one or several inference procedures. A primitive was associated to each primary knowledge area. The selection of appropriate primitives to support a knowledge model is an important step within the development of the application where the developer establishes a connection between the conceptual description and the computational support. KSM has a library of primitives of representation that can be extended with other more domain specific primitives. Note that the use of a library provides a multi-representation environment where developers can select the most appropriate technique for each module of the architecture. This is particularly important in real systems where efficiency must be taken into account.

Primary knowledge area	Abstraction	Problem types	Impact categories	Future demand	Risk balance criteria	System model	Action types
Primitive of represen-tation	Functional + temporal represent.	Frames with uncertainty	Bayesian network	Rules	Rules	Dynamic bayesian network	Rules

Figure 7: *Computational support of primary knowledge areas with primitives of representation.*

For the case of the knowledge model described in this article, figure 7 shows the primitives associated to each primary area. We tried to use domain-independent primitives in order to keep the maximum degree of generality and reusability of the knowledge model. Here, knowledge-based techniques for knowledge representation such as rules, frames or bayesian networks were very appropriate given that they provide a good level of flexibility to be applied to different domains. In particular, for example, for the case of the system model, dynamic bayesian networks were used to represent the causal influences in the different points of the river. For example figure 8 shows the knowledge representation followed with bayesian networks to describe the behavior of a reservoir. More details about this can be found at (Molina et al., 05).

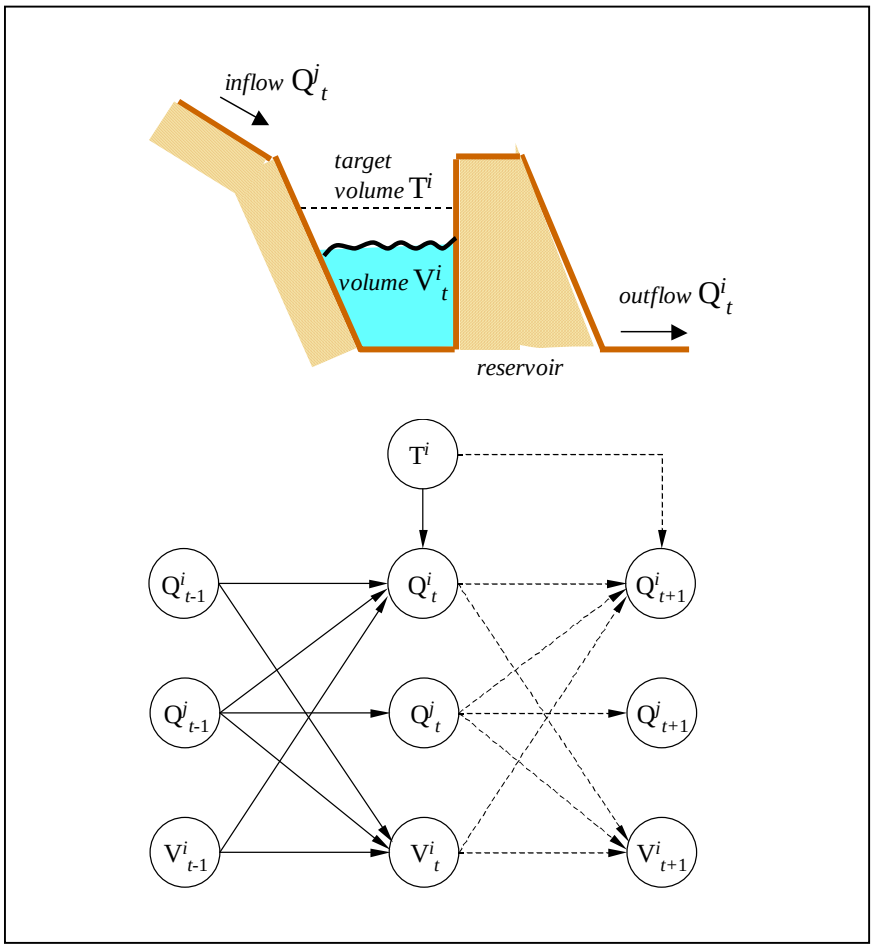

Figure 8: *Example of knowledge representation with bayesian network followed to describe the behavior of a reservoir.*

These structures (one for each agent) that include knowledge areas, tasks, methods and vocabularies together with the primitives constitute patterns that are general and reusable. They are called *generic models* and they are abstract descriptions of the types of knowledge and strategies of reasoning for each type of agent. To develop a model for a particular basin it is necessary to create a specific models as instances and extensions of the generic models. We developed two different specific models corresponding to two specific basins: (1) the Júcar basin and (2) the basins of the South of Spain. The generic model was used as a template and completed with the following information:

- *Specific agents*. For each type of agent (reservoir management agent, problem detection agent, hydraulic agent and protection agent) several instances were defined. For example, *Conde de Guadalhorce* and *Limonero* are cases of reservoir management agents in the basin of the South of Spain. Five reservoir agents were defined for the South of Spain and four reservoir agents for the case of the Júcar river (in total 23 agents were defined for the Júcar river).
- *Specific vocabularies*. For each agent, a specific vocabulary was defined as instance of generic vocabularies. This was done in KSM using the Concel language with which specific concepts (e.g., the particular pluviometers, flow sensors, etc. in the Júcar River) are defined as subclasses and instances of more general concepts defined in generic vocabularies.
- *Specific knowledge bases*. The particular content of each knowledge base was written using the language provided by primitives. For instance, particular bayesian networks, rule bases, etc. were created and formulated with the specific languages provided by the corresponding primitives.

Figure 9 shows a summary of the operations done with the help of the facilities provided by KSM to construct the complete model. KSM provides graphical editors to construct the structure of both the generic and specific models for each agent. KSM also provide graphical editors that allow the user to select primitives of representation from a library and to associate them to primary areas. KSM also provides local text editors to write methods for tasks (in Link language), generic and specific vocabularies (in Concel language) and knowledge bases with the specific languages provided by the primitives.

In order to organize the multiagent architecture, it was necessary to design a particular solution for SAIDA using KSM (Molina, Cuena, 04). Figure 10 shows the final global architecture with KSM adopted by the SAIDA system, where efficiency was an important factor. In this architecture, there are several copies of the KSM tool, where each one supports a family of agents with the same generic model. For example, the left hand side of the figure shows the solution for reservoir management agents. Here, a copy of the KSM tool serves as a software platform where it is installed a library of primitives of representation that are used to implement the generic knowledge model for reservoir management agents. In its turn, this structure of generic model is shared by the particular knowledge models of each reservoir management agent that includes specific knowledge bases. This organization is similar to the other types of SAIDA agents: problem detection agents, civil protection agents and hydraulic agents. In addition to that, a global mechanism is used to communicate agents according to the required individual autonomy in the model.

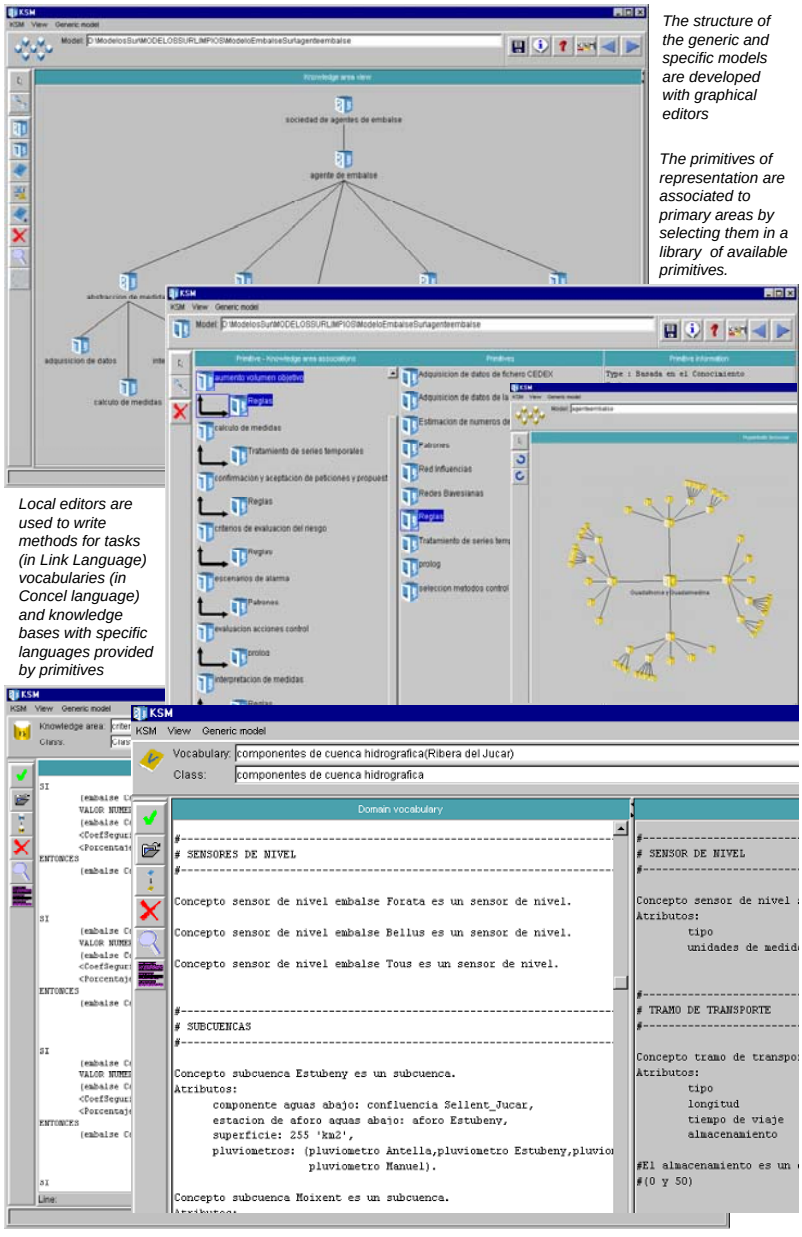

Figure 9: *Summary of facilities provided by KSM to build the knowledge model.*

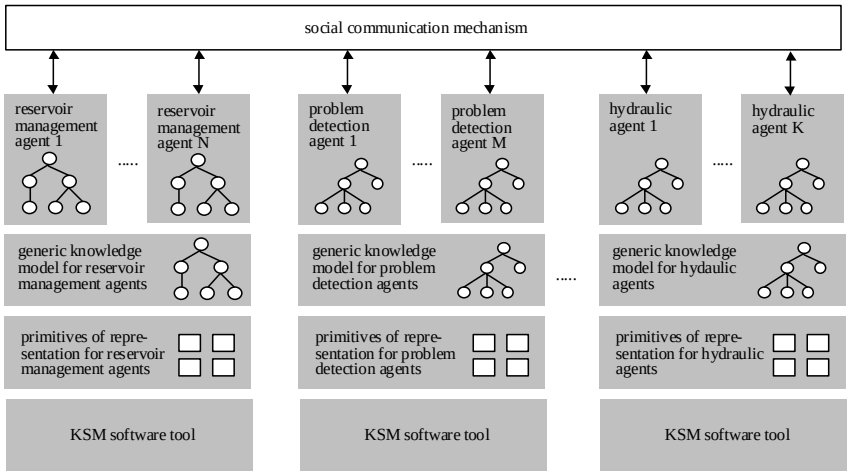

Figure 10: *Distributed organization using KSM to support the multiagent architecture of the SAIDA system.*

5. Discussion

Following the approach described in this article, two main decision support systems were developed for two different basins in Spain corresponding to (1) the Júcar basin and (2) the basin of the South of Spain. Details about these models and its evaluation can be found at (Molina, Blasco, 03; Molina et al., 05). These two realizations showed satisfactory results of the final applications which demonstrated the feasibility of the technical solution. Different decision support systems in other domains were also developed following the same approach using the KSM tool with successful results (see for example (Molina et al., 98a) for a system in the domain of traffic control).

Other tools similar to KSM are KREST (Steels, 92) that followed the concept of *components of expertise* (a similar approach of the task-method-domain approach) and PROTÉGÉ-II (Puerta et al., 93) which the last years has been successfully oriented to the development of ontologies. KSM differs from KREST mainly in the grain size of reusable components, which are more complex in KSM. KREST was developed for representations based on the Lisp language while KSM is able of integrating components of different languages. KSM shares with the original version of PROTÉGÉ-II the idea of reusable component (mechanism in PROTÉGÉ-II) although PROTÉGÉ-II did not provided a solution to create complex architectures by integration of several mechanisms.

Using knowledge modeling tools like KSM provides certain advantages in the development of a DSS in complex real world problems that require knowledge representation solutions. This advantages can be considered in the different phases of the development of the system: analysis, design, implementation and maintenance. During the *analysis* phase, a tool like KSM follows a modeling approach using certain high level description entities (e.g., tasks, methods, knowledge areas, etc.) that facilitate the homogeneous description of the system and help to cope with the complexity of the system. This more natural description can be used to present the model to end-users who may verify the proposed architecture before starting its implementation, which facilitates early validation of the system. In addition to that, the description uses standard components (tasks, methods, etc.) followed by different knowledge engineering methodologies, which facilitates the communication between professionals in this area.

The *design* and *implementation* of the models normally require efficient solutions to keep a uniform integration of different symbolic languages. For this purpose, KSM uses the modeling concept called *knowledge area* that provides encapsulation for tasks, methods, vocabularies and knowledge bases. In addition to that, KSM provides an open library of primitives of representation, each one with its particular symbolic language, that can be associated to knowledge areas to select the most adequate representation for each case. Thus, for example, in the knowledge model of the SAIDA system, there are knowledge areas such as *impact categories*, *future demand* and *problem types* that are respectively supported by three different primitives of representation: bayesian networks, rules and frames with uncertainty.

SAIDA follows a multi-agent approach to organize the whole components of its architecture. In this case, KSM was very useful to formulate the generic model for each type of agent. Then, each generic description was used as template to formulate the particular knowledge model for each specific agent. Thus for instance, with KSM a generic model was formulated for the reservoir management agent with a structure of knowledge areas, tasks, methods, generic vocabularies and primitives of representation. Then, this generic structure was used as template to construct the model for five particular reservoir in the case of the basins of the South of Spain (for example, the reservoir of *Conde de Guadalhorce*). For each specific agent, specific vocabularies and specific knowledge bases were written. Thus, although KSM does not directly support a multiagent organization, KSM is especially useful to formulate general models that are shared by specific occurrences of agents.

During the *maintenance*, the developer may use KSM to consult the structure of the conceptual model of the final DSS and she or he may access to local knowledge bases following this structure. The role of KSM in this phase is to allow the user to open the final DSS to consult the knowledge that supports the reasoning. Thus, with KSM, the DSS is not perceived as a black-box but, on the contrary, the system is open to be adapted it to new requirements. This can be done at two levels (1) by accessing and modifying the content of the knowledge bases and vocabularies using the declarative languages (rules, frames, etc.) and (2) by modifying the conceptual

structure of knowledge areas using the graphical editors and the Link language provided by KSM. The experience of SAIDA during maintenance revealed that the final software architecture was flexible to accept a number of changes easily thanks to the use of KSM. This was done mainly by technicians in programming languages although other specialists in hydrology (non-programmers) were also able to use KSM to modify the knowledge models. However, the experience showed that a complete and extensive maintenance of the SAIDA knowledge models by non-programmers still requires additional tools for knowledge acquisition. This need constitutes currently one of our lines of research for which some results have been already produced (Molina, Blasco, 04).

In summary, a knowledge modeling tool like KSM implies a significant change compared to the traditional view of computer system development. The developer who uses KSM conceives the development of a computer system as an activity of knowledge modeling. This knowledge model is not formulated by programming a set of instructions following the classical procedural approach. Instead of that, the model is constructed as an activity of selecting, adapting and assembling reusable high level components following a knowledge-based approach which is more intuitive and natural and makes it very appropriate to be used in the development of complex systems as it may happens in DSS with knowledge-based techniques.

6. Conclusions

The case presented in this article illustrates how knowledge modeling tools can be very useful in the development of complex decision support systems. The paper shows the example of the decision support system SAIDA in the domain of hydrology that was developed using the KSM tool.

KSM was especially useful in the development of the SAIDA system because it provided (1) a particular modeling approach to cope with the complexity of the knowledge of the system and (2) a library of preprogrammed building blocks, called primitives of representation, that helped to efficiently construct the final operational version of SAIDA using the most adequate knowledge representation for each case. KSM was also useful during the maintenance of SAIDA given that KSM facilitated the access to the knowledge organization to consult and modify both the content of knowledge bases and the structure of the model.

The generality of the modeling approach provided by KSM was demonstrated with the development of decision support systems in other fields. For instance, KSM was successfully applied in the domain of road traffic control following the approach presented in this paper with two different realizations for the cities of Barcelona and Madrid (Molina et al., 98a).

Acknowledgements.

The development of the SAIDA system was mainly supported by the Ministry of Environment of Spain (*Dir. General de Obras Hidráulicas y Calidad de las Aguas*) with the participation of local public organizations from river basins (*Conf. Hidrográfica del Júcar* and *Conf. Hidrográfica del Sur de España.)*. This research was also possible with funding provided by the Ministry of Education and Science of Spain (*E-VIRTUAL* and *RIADA* projects).

References

Brown D., Chandrasekaran B.: Design Problem-solving: Knowledge Structures and Control Strategies, Morgan Kaufman, 1989.

Chandrasekaran, B.: “Towards a Taxonomy of Problem Solving Types” A.I. Magazine 4 (1) 9-17, 1983.

Chandrasekaran, B.: "Generic Tasks in Knowledge Based Reasoning: High Level Building Blocks for Expert Systems Design" IEEE Expert, 1986.

Clancey W.: “Heuristic Classification”. Artificial Intelligence 27, 1985.

Cuena, J., Molina M.: “KSM: An Environment for Knowledge Oriented Design of Applications Using Structured Knowledge Architectures” in Applications and Impacts. Information Processing IFIP’94, Volume 2. K. Brunnstein y E. Raubold (eds.) Elsevier Science B.V. (North-Holland). 1994.

Cuena J., Molina M.: “The Role of Knowledge Modeling Techniques in Software Development: A General Approach based on a Knowledge Management Tool”. International Journal of Human-Computer Studies. No. 52. pp 385-421. Academic Press, 2000.

Marcus S., McDermott J.: “SALT: A knowledge acquisition language for propose-and-revise systems”. Artificial Intelligence, 39(1) 1-38, 1989.

McDermott, J.: “Preliminary Steps Toward a Taxonomy of Problem Solving Methods” in Automating Knowledge Acquisition for Expert Systems, S.Marcus ed., Kluwer Academic, Boston, 1988.

Molina M.: “Desarrollo de Aplicaciones a Nivel Cognitivo Mediante Entornos de Conocimiento Estructurado” PhD Thesis, Department of Artificial Intelligence. Universidad Politécnica de Madrid. November, 1993.

Molina M., Blasco G: “A Multi-agent System for Emergency Decision Support”. Proceedings of the Fourth International Conference Intelligent data Engineerging and automated Learning IDEAL 2003. LNCS, Springer. 2003.

Molina M., Blasco G: “KATS: A Knowledge Acquisition Tool Based on Electronic Document Processing”. 14th International Conference on Knowledge Engineering and Knowledge Management EKAW 04. In "Engineering Knowledge in the Age of the Semantic Web" Lecture Notes in Artificial Intelligence 3257. Springer Verlag. Whittlebury Hay, UK, October 2004.

Molina M., Cuena J.: “Using Knowledge Modelling Tools for Agent-based Systems: The Experience of KSM”. In J.Cuena et al. (eds.) “Knowledge Engineering and Agent Technology” IOS Press, Amsterdam, 2004.

Molina M., Fuentetaja R., Garrote L.: “Hydrologic Models for Emergency Decisión Support Using Bayesian Networks”. 8th European Conference on Symbolic and Quantitative Approaches to Reasoning with Uncertainty, ECSQARU 05. In “Symbolic and Quantitative Approaches to Reasoning with Uncertainty”. Lecture Notes in Artificial Intelligence LNAI 3571. Barcelona, July, 2005.

Molina M., Hernández J., Cuena J. M.: “A Structure of Problem-solving methods for real-time decision support in traffic control”. International Journal of Human-Computer Studies. No. 49. pp 577-600. Academic Press, 1998a.

Molina M., Sierra J., Serrano J. M.: “A language go formalize and to operationalize problem solving strategies of structured knowledge models”. 8th Workshop on Knowledge Engineering: Methods and Languages. KEML 98. Karlsruhe, Germany, 1998b.

Molina M., Sierra J., Cuena J.: "Reusable Knowledge-based Components for Building Software applications: A Knowledge Modelling Approach" International Journal of Software Engineering and Knowledge Engineering, 1999.

Puerta A., Tu S.W., Musen M.A.: “Modeling Tasks with Mechanisms”. International Journal of Intelligent Systems, Vol 8, 1993.

Schreiber G., Akkermans H., Anjewierden A., De Hoog R., Shadbolt N., Van de Velde W., Wielinga B.: “Knowledge engineering and management. The CommonKADS methodology” MIT Press, 2000.

Steels, L.: "Components of Expertise" AI Magazine, Vol. 11(2) 29-49. 1990.

Steels L.: “Reusability and Configuration of Applications by Non-programmers”. Proc. Artificial Intelligence form the Information Processing Perspective (Workshop AIFIPP 92). Madrid, 1992.

Wielinga B.J., Schreiber A.T., Breuker J.A.: "KADS: A Modelling Approach to Knowledge Engineering". Knowledge Acquisition, 1992.