K-DSS: A Decision Support System for Identifying and Evaluating Crucial Knowledge

Inès Saad, Camille Rosenthal-Sabroux and Salem Chakhar
LAMSADE laboratory, University of Paris Dauphine,
Place du Maréchal de Lattre de Tassigny, 75775 Paris Cedex 16, Paris, France.
Telephone: +33 (1)-44-05-41-20. Fax: +33 (1)-44-05-41-11.
Email: {saad, sabroux, chakhar}@lamsade.dauphine.fr

***Abstract*—The objective of this paper is to introduce K-DSS, a Decision Support System for identifying and evaluating crucial Knowledge. K-DSS is an implementation of a two phases-based methodology conducted and validated in the PSA Peugeot Citroën automobile company in France. Attention is especially devoted to present the conceptual and functional architectures of K-DSS. The implementation of K-DSS is also addressed.**

***Index Terms*—Crucial knowledge, Decision support system, Knowledge capitalizing, Knowledge management.**

I. INTRODUCTION

CAPITALIZING on the company's knowledge is increasingly being recognized. Capitalizing on all the company's knowledge requires an important human and financial investments. To optimize the capitalizing operation, one should focalize on only the so called "crucial knowledge", that is, the most valuable knowledge. This permits particularly to save time and money.

In practice, decision makers use tacit and explicit knowledge available in various forms (e.g. decision support system, knowledge-based system, database, documents) in the organization to select, from a set of options, the alternative(s) that better response(s) to the organization objectives. The main objective of capitalizing is to extract tacit knowledge [19], that are not explicitly defined and which are considered crucial for improving decisions and their outcomes [6]. As mentioned in [18], "tacit knowledge is quite beneficial to a faster decision making process". Thus, companies should invest in engineering methods and tools in order to preserve the knowledge, especially of tacit nature, related to the decision making process. K-DSS, a Decision Support System for identifying and evaluating crucial Knowledge, is one of such tools.

Most importantly, K-DSS is an implementation of a two phases-based methodology conducted and practically validated in the PSA Peugeot Citroën automobile company in France. More specifically, we have focalized on the FAP (for Particle-based Filter) development projects: FAPx, FAPy, FAPz and FAPw (x, y, z and w denote the successive generation of FAP system). FAP is a depollution sub-system integrated in the exhaust system. The objective of PSA Peugeot Citroën company is to transfer the knowledge developed in FAPx for use:

- with other types of vehicles;
- with projects concerned with definition of the new systems of FAP (i.e. FAPy, FAPz et FAPw).

The objective of this paper is to describe K-DSS. Attention is especially devoted to present the conceptual and functional architectures of K-DSS. The implementation of K-DSS is also addressed.

The remain of the paper goes as follows. Section II very briefly introduces the proposed methodology. Section III presents the conceptual architecture of K-DSS. Section IV describes the functional architecture of K-DSS. Section V provides a brief description of the developed system. Section VI concludes the paper.

II. METHODOLOGY

The adopted methodology is composed of two phases. A detailed description of it is available in [22]. The first phase is relative to constructive learning devoted to infer the preference model of the decision makers. Practically, it consists in inferring, through the DRSA (Dominance-based Rough Set Approach) [8] method—which is an extension of rough set theory [20] and which is devoted to multi-criteria sorting problems—of a set of decision rules from some holistic information—in terms of assignment examples—provided by the decision makers. This phase includes also the identification, using GAMETH (Global Analysis METHodology) framework [10], of a set of "knowledge of reference" and their evaluation with respect to a convenient set of criteria.

Inspiring from the systemic approach of [17] and by using the bottom-up approach, three sub-families of criteria where constructed: (i) *knowledge vulnerability family* that are devoted to measure the risk of knowledge lost and the cost of its (re)creation; (ii) *knowledge role family* that are used to measure the contribution of the knowledge in the project objectives. Each criterion of this family corresponds to an objective; and (iii) *use duration family* that is devoted to measure the use duration of the knowledge basing on the company average and long term objectives.

To evaluate each knowledge K_i in respect to the each objective O_j, we have developed the computing model illustrated in Figure 1. As it is shown in this Figure, the computing model is

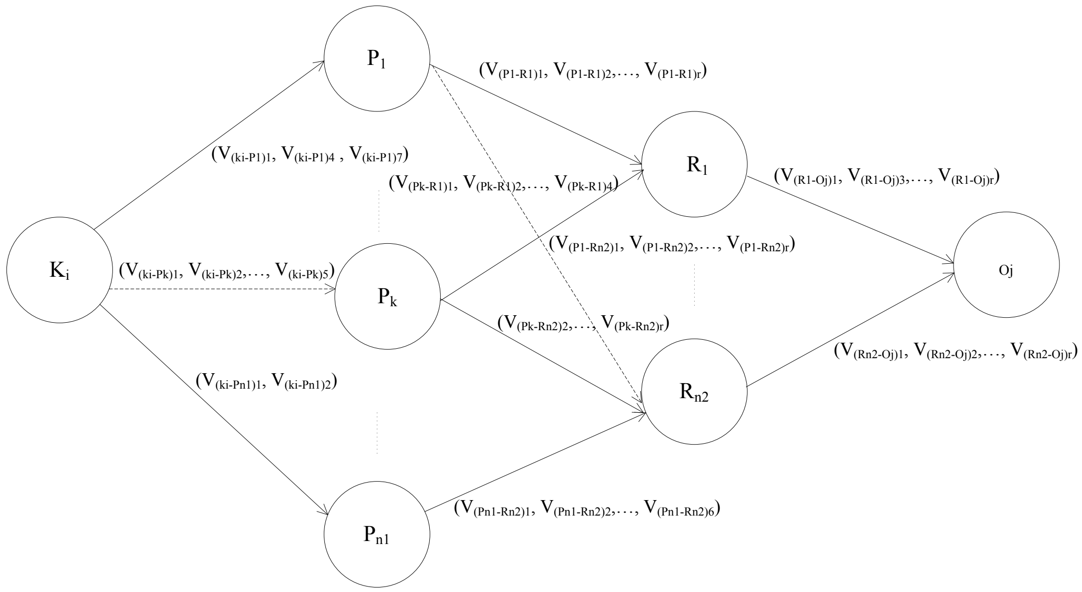


Fig. 1. Contribution degree computing model

an oriented four levels graph. The first level corresponds to the piece of knowledge K_i to be evaluated. The second level corresponds to processes $P_1 \cdots P_{n_1}$; n_1 is the number of processes. The third level corresponds to projects $R_1 \cdots R_{n_2}$; n_2 is the number of projects. The last level corresponds to the objective O_j. The valuation $V_{(K_i-P_k)1}, \cdots, V_{(K_i-P_k)r_\alpha}$ of the vertex (K_i, P_k) is provided by r_α decision makers. The valuation $V_{(P_k-R_z)1}, \cdots, V_{(P_k-R_z)r_\beta}$ and $V_{(R_z-O_j)1}, \cdots, V_{(R_z-O_j)r_\gamma}$ correspond to vertexes (P_k, R_z) and (R_z, O_j), respectively. Note that the dimension of valuation vectors is not static and vary along with the number of decision makers (denoted above as r_α, r_β and r_γ for the first, second and third level, respectively) which are able to evaluate the considered vertex. The evaluations of knowledge in respect to families (i) and (iii) are provided by the decision maker(s).

Once all knowledge are evaluated in respect to all criteria, the next step is an iterative procedure permitting to conjointly infer the decision rules. Two decision classes have been defined: $Cl1$: “non crucial knowledge” and $Cl2$: “crucial knowledge”. This procedure is composed of four substeps. Basing on the set of “knowledge of reference” and the decision classes, the first substep consists to determine with each decision maker the assignment of these “knowledge of reference” into the decision classes $Cl1$ and $Cl2$. The second substep permits to infer a set of decision rules for each assignment example determined in the previous substep. The third substep consists to modify the assignment examples or the evaluation with the concerned decision maker. This substep is an iterative one and is devoted to resolve inconsistency problems. Finally, we identify, with the help of the decision makers, a subset of collectively accepted decision rules.

In the second phase, the analyst uses the preference models of the different stakeholders defined in the first phase to assign new knowledge, called “potential crucial knowledge”, to the classes $Cl1$ or $Cl2$. More specifically, a multi-criteria classification of “potential crucial knowledge” is performed on the basis of the decision rules that have been collectively identified by the decision makers in the first phase. The generated “potential crucial knowledge” are analyzed and then evaluated against the criteria identified in the first phase. Then, they are assigned to one of two decision classes $Cl1$ or $Cl2$. Finally, we remark that the methodology was developed and validated within real-world data in the PSA Peugeot Citroën company but it is generic enough that may be easily conduced within other similar companies.

III. Architecture of K-DSS

As for most of DSS, K-DSS contains four main components: (i) *graphical interface*; (ii) *model base* which is the repository of all the algorithms need to implement the proposed methodology; (iii) *database* which is the repository of data and eventually the parameters needed for executing the algorithms; and (iv) *knowledge base* which is the repository of all the pieces of knowledge represented in terms of facts and rules.

A. Graphical interface

The graphical interface defines how the different resources of K-DSS (algorithms, database, knowledge base) are used. The interface of K-DSS is based on the GUI (Graphical User Interface) environment, i.e., an hierarchy of menus and sub-menus offering to the user transparency, simplicity and conviviality in the exploitation of the system.

B. Model base

The model base of K-DSS regroups all the algorithms required to implement the methodology. More specifically, it contains: (i) the algorithms for computing the contribution degrees of the knowledge into the objectives, and (ii) the algorithms used to infer decision rules.

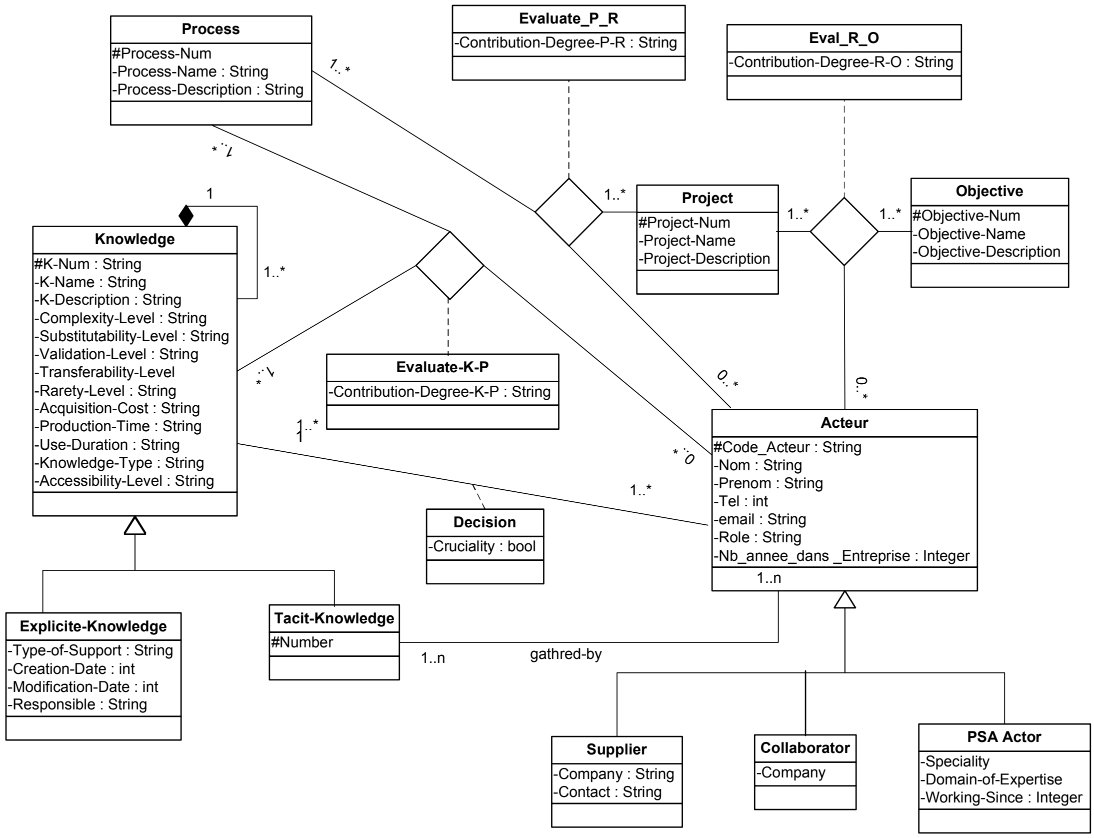


Fig. 2. UML-based conceptual schema of the database

1) Algorithms for computing contribution degrees: The system contains the three following algorithms:

- $Max_{p \in P}\ Min_{e \in p}\ Min_{d \in D}\ v^d(e)$
- $Max_{p \in P}\ Min_{e \in p}\ Median_{d \in D}\ v^d(e)$
- $Max_{p \in P}\ Min_{e \in p}\ Max_{d \in D}\ v^d(e)$

where P is the set of paths from K_i to O_j; p is a path from the set of paths P; $D = \{d_1, \cdots, d_r\}$ is the set of decision makers; and $v^d(e)$ is the evaluation of the vertex e from path p designing the contribution degree of a knowledge to a process, a process to a project or a project to an objective, according to decision maker d.
In our study, the responsibles of FAP project have privileged the first algorithm. Due to the innovative nature of the FAP development project, the decision makers show a risk-averse behavior by taking on each vertex the highest value in order to maximize the contribution. However, the decision makers show a less risk-averse attitude for well established projects. In this case, the decision makers prefer to take on each vertex the lowest value. They can also adopt a neutral behavior by taking on each vertex the median value. Incorporating these three algorithms into the system enhances the flexibility of K-DSS by offering to decision makers the possibility to select, with the help of the analyst, the most appropriate algorithm. Note that other algorithms may be added to the system.

2) Algorithms for the inference of decision rules: The model base contains two algorithms for decision rules induction. Generally, the induction algorithms permit to produce either (i) a minimal covering set of decision rules, i.e., a subset of non-redundant and complete decision rules as for example the DOMLEM (see [9]) algorithm; or (ii) a set containing all the decision rules as for example the algorithms LEM2 (Learning from Examples Module, version 2; which is a part of the data mining system LERS—Learning from Examples based on Rough Sets—; see [12], [13]) or Explore (see [25]). Here, we have used the DOMLEM and Explore algorithms. These two algorithms use the rough set theory [20].

C. Database

The UML-based conceptual schema of the database is shown in Figure 2. The central class in the model is the class “Knowledge”. It is described with an unique number (K-Num), a name (K-Name), a description (K-Description), eight attributes (Complexity-Level, Substitutability-Level, Validation-Level, Transferability-level, Rarety-Level, Acquisition-Cost, Production-Time, Accessibility-Level) corresponding to the eight criteria $g_1, \cdots, g_8$ composing *knowledge vulnerability family*, use duration (Use-Duration) corresponding to the only criterion, g_{15}, of *use duration family*, (Knowledge-Type) (i.e. “knowledge of reference” or “potential crucial knowledge”). Note finally that a piece of knowledge may be composed of several elementary knowledge. This is enhanced with the aggregation relation defined on the class “Knowledge”.
The classes “Explicit-Knowledge” and “Tacit-Knowledge”

are specializations of the class “Knowledge”. The “Explicit-Knowledge” class permits to identify for each explicit knowledge the set of supports (documents, database, knowledge base system) on which this knowledge is inscribed. If the knowledge is tacit, it is characterized with the person who gathers it. This information is deduced from the relationship “Gathers-By” between “Tacit-Knowledge” and “Actor”. The class “Actor” contains the information relative to the different actors (Name, Telephone, Email, Role, Service-Length). The class “Actor” is specialized into three classes: “Supplier”, “Collaborator” and “PSA Actor”.

The three classes “Process”, “Project” and “Objective” permit to handle the information relative to the names and descriptions of processes, projects and objectives, respectively. The association class “Evaluate-K-P” between “Actor”, “Process” and “Knowledge” stores the contribution degree of a knowledge into a process (Contribution-Degree-K-P) attributed by a given actor.

As it is illustrated in Figure 2, an actor evaluate zero, one or many knowledge regarding one or many processes. The association classes “Evaluate-P-R” between the classes “Process”, “Project” and “Actor”; and “Evaluate-R-O” between the classes “Objective”, “Project” and “Actor” store the contribution degrees “Contribution-Degree-P-R” and “Contribution-Degree-R-O”, given by an actor to mesure the contribution of a process into a project; and of a project into an objective, respectively.

As shown in Figure 2, an actor evaluates one or many processes according to one or many projects. Similarly, it evaluates one or several projects according to one or several objectives. For a given project and a given process, it exists zero, one or several evaluations provided by zero, one or several actors. This is also true for a given project and a given objective.

The association class “Decision” contains the decision given by an actor concerning a given knowledge. According to the model of Figure 2, an actor assigns one or several knowledge to the classes $Cl1$ or $Cl2$. A given knowledge can not be assigned to different categories for the same decision maker. Due the fact that the same knowledge may be evaluated by different actors, the creation of class “Decision” is necessary.

D. Knowledge base

To construct the knowledge base, we have used the expert systems generator JESS (Java Expert System Shell[1]). Since we are interested only with crucial knowledge, the rules base contains only the rules permitting to assign with certainty “potential crucial knowledge” to the class “$Cl2$: crucial knowledge”. This because in our application only two classes have been defined and the rules relative to the class “crucial knowledge” will be redundant. However, if several classes have been defined, we should maintain all the rules.

A rule in JESS is defined through the function **defrule**. An example relative to our application is given in Figure 3. The fact base contains the initial facts relative to knowledge of reference issued from the decision table. A fact in JESS is defined through the function **defacts**. Figure 4 gives a JESS definition of a fact relative to the application.

[1]JESS is a free package, which is available on http://herzberg.ca.sandia.gov/jess/

(**defrule** *rule1*
 (Knowledge (K-Num ?K)
  (K-Description ?KD)
  (K-Name ?KN)
  (K-Description ?KD)
  (Complexity-Level ?CL)
  (Substitutability-Level ?SL)
  (Validation-Level ?VL)
  (Transferability-level ?TL)
  (Accessibility-Level ?AL)
  (Rarety-Level ?RL)
  (Acquisition-Cost ?AC)
  (Acquisition-Time ?PT)
  (Use-Duration ?UD)
 => (printout outfile “crucial knowledge”)
)

Fig. 3. An example of a rule definition

(**defacts** *knowledge*
 (Knowledge (K-Num K_1)
  (K-Name *knowledge relative to additive dosage*)
  (K-Description)
  (Complexity-Level *complex*)
  (Substitutability-Level *substitutable*)
  (Validation-Level *experimental*)
  (Transferability-level *hardly transferable*)
  (Accessibility-Level *easy*)
  (Rarety-Level *rare*)
  (Acquisition-Cost *low*)
  (Acquisition-Time *high*)
  (Use-Duration *high*)
)

Fig. 4. An example of a fact definition

IV. FUNCTIONAL ARCHITECTURE OF K-DSS

Figure 5 describes the functional architecture of K-DSS. Two phases may be distinguished in this figure. The first phase is relative to the construction of the preference model. The preference model is represented in terms of decision rules. The second phase concerns the classification of potential crucial knowledge by using the rules collectively identified (by all the decision makers) in the first phase.

A. Phase 1. Construction of the preference model

The first step consists in identifying, from the ones proposed, an algorithm for computing the contribution degrees. The selection is collectively established by all the decision makers with the help of the analyst. Whatever the selected algorithm, it uses the matrices Knowledge-Process (K-P), Process-pRoject (P-R) and pRoject-Objective (R-O) extracted from the database—more specifically from the three association classes “Evaluate-K-P”, “Evaluate-P-R ” and “Evaluate-R-O ”—to compute the contribution degree of each piece of knowledge into each objective. To avoid data redundance, these matrices are not explicitly stored in the database but generated during processing. Only their intentional definitions are permanently stored in the system.
Once these matrices are generated, the contribution degrees are first stored (temporally) in a decision table and then introduced in the database. The structure of the decision table is shown in Figure 6. As for matrices, only the intentional definition of the decision table is maintained in the system.
The decision table contains also the evaluation of the “knowledge of reference” concerning the vulnerability and use duration criteria extracted from the database (from the class “Knowledge” precisely). These evaluations are collectively defined and introduced by the analyst in the database. The analyst should introduce in the decision table, and for each decision maker k, the decisions concerning the assignment of “knowledge of reference” into the classes $Cl1$ and $Cl2$.
The decision table contains, in addition to the columns relative to vulnerability and those relative to contribution degree and use duration criteria, as many columns as decision makers. Once the decision table is generated, it will be used as the input of the induction algorithm selected by the decision makers (DOMLEM or Explore). This algorithm permits to generate the list of the initial rules for each decision maker k. It is important to mention again that only rules relative to class $Cl2$ are stored. Then, each decision maker should select a subset from these initial rules. The next step in this phase consists to collectively select, from the set of decision rules individually identified by the different decision makers, a subset of decision rules that will be used latter by JESS for the classification phase. Note that the rules generated by DOMLEM or Explore are in text format; they are automatically traduced into a format compatible with the one of JESS and then stored in the rule base.

Knowledge of reference	Criteria			Decision
	g_1	$\cdots$	g_m	
K_1	$f(K_1, g_1)$	$\cdots$	$f(K_1, g_m)$	C1/C2
$\cdots$	$\cdots$	$\cdots$	$\cdots$	$\cdots$
K_n	$f(K_n, g_1)$	$\cdots$	$f(K_n, g_m)$	C1/C2

Fig. 6. Decision table for one decision maker

B. Phase 2. Evaluation of potential crucial knowledge

The second phase consists in classifying the new knowledge called “potential crucial knowledge”. As the previous one, this phase starts by identifying the algorithm to use to compute

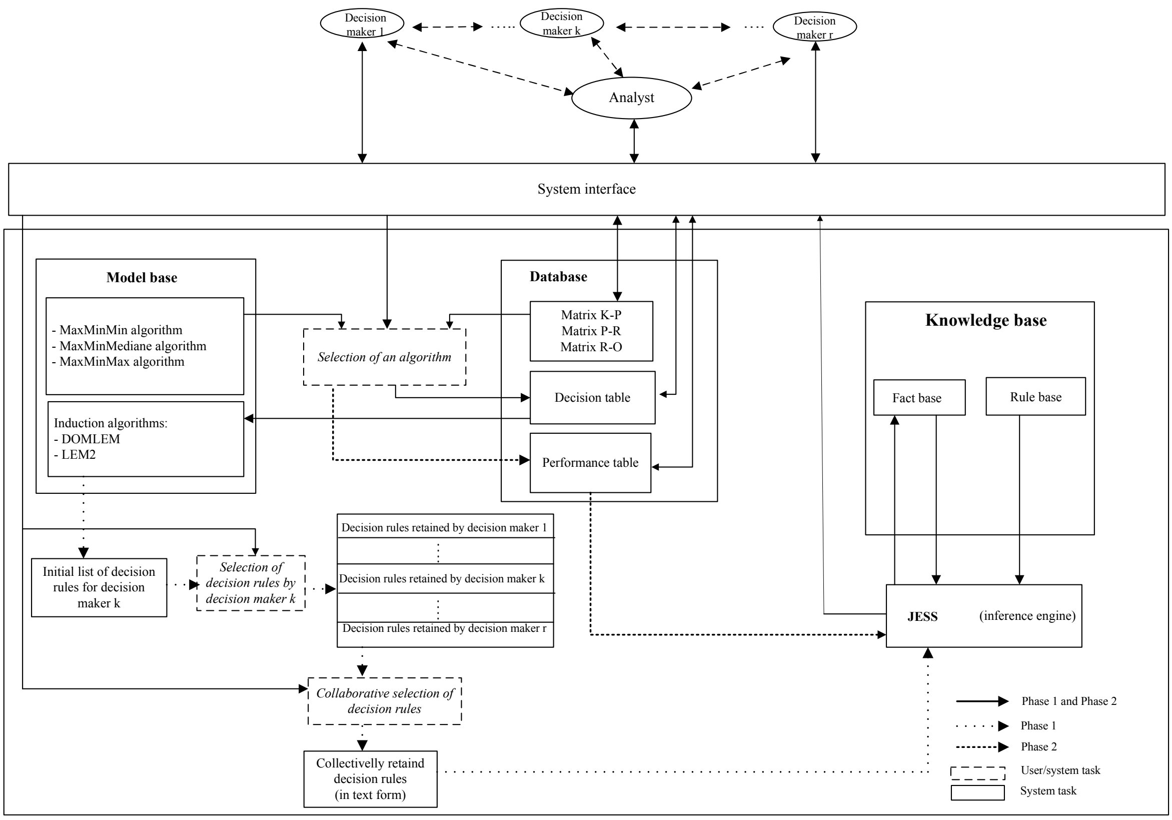


Fig. 5. Functional architecture of K-DSS

the contribution degree of each piece of knowledge into each objective. This algorithm uses as input the information relative to the performances of potential crucial knowledge previously introduced in the matrices K-P, P-R and R-O. The results are stored in a performance table. The structure of the performance table is shown in Figure 7. The information contained in the performance table are then transformed into facts. The inference engine incorporated in JESS verifies first if exists at least one rule (in the rule base) that verifies the different facts and if this holds, the knowledge is classified as crucial; otherwise the piece of knowledge is considered non crucial. An update of rule and fact bases any time a fact is verified by at least one rule is performed.

Potential crucial knowledge	Criteria		
	g_1	$\cdots$	g_m
K_1	$f(K_1, g_1)$	$\cdots$	$f(K_1, g_m)$
$\cdots$	$\cdots$	$\cdots$	$\cdots$
K_n	$f(K_n, g_1)$	$\cdots$	$f(K_n, g_m)$

Fig. 7. Performance table

V. IMPLEMENTATION

In section we provide a brief description a prototype implementing K-DSS. K-DSS was implemented with Visual Basic. The user can use the different capabilities of GUI interfacing system to, among others, introduce required data, infer decision rules, classify knowledge into $Cl1$ or $Cl2$.
Figures 8, 9 and 12 presents three printed screens from K-DSS. The screen in Figure 8 permits to generate Matrix K-P containing the evaluation of each knowledge in respect to each process. As it is shown in this screen, the user selects the piece of knowledge to evaluate and then introduces the evaluation directly or by selecting the desired evaluation from the provided drop-down list. The user may also add/remove a process from the list initially shown. Similar interfaces are used for process-project and project-objectives evaluations. They permit to generate Matrix P-R and Matrix R-O, respectively.

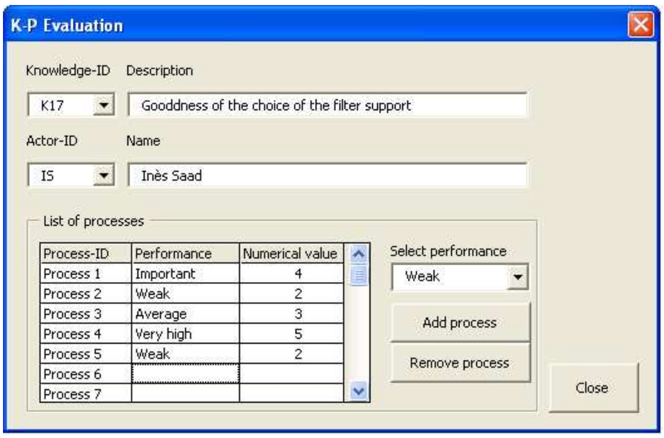


Fig. 8. Knowledge-Process evaluation interface

Once all the data are introduced, the user may use the interface shown in Figure 9 to compute the contribution degrees of each knowledge to each objective. First, s/he should select the computing algorithm. As mentioned earlier, three algorithms are provided by the system: (i) $Max_{p \in P}\ Min_{e \in p}\ Min_{d \in D}\ v^d(e)$, (ii) $Max_{p \in P}\ Min_{e \in p}\ Median_{d \in D}\ v^d(e)$; and (iii) $Max_{p \in P}\ Min_{e \in p}\ Max_{d \in D}\ v^d(e)$. Figure 10 provides the general schema of the $MaxMin$ algorithm, which is the common part to the three computing algorithms. Minor modifications are required to implement the three algorithms.

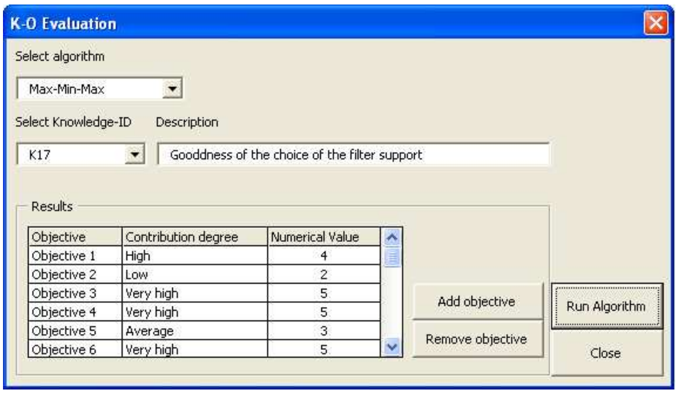


Fig. 9. Contribution degree computing interface

To infer decision rules, we have used JESS. To incorporate JESS in our system, we have developed an executable file (*inference.exe*) in JAVA to import JESS DLLs (see Figure 11). K-DSS and JAVA dialogue is completely transparent to users. As shown, in Figure 11, K-DSS automatically generates an input text file (*input.txt*) which in used by *inference.exe*. The results generated by JAVA are then stored, by *inference.exe* in an output text file (*outpout.txt*). This last one is then used by K-DSS to provide results (in terms of decision rules) to the user.
The decision rules are first generated by DOMLEM or Explore. These rules are initially expressed in the following mathematical form:

If $f(x, g_3) \geq 2 \wedge f(x, g_6) \geq 2 \wedge f(x, g_{12}) \geq 4 \wedge f(x, g_{15}) \geq 2$
Than $x \in Cl2$

The rules are automatically traduced, by K-DSS, to apply to the syntax of JESS. For example, the rule cited above will be traduced as follows:

IF K_i.Substituable-Level is "at least weak" and
K_i.Rarety-Level is "at least rare" and
K_i.Competitivity is "at least high" and
K_i.use-duration is at least "average"
THEN K_i is at least in Cl2

This rule means that knowledge K_i is considered to be crucial (i.e. K_i belongs to the class $Cl2$), if it is difficult to replace it, it is scares, has an important impact on commercial position of the company and with a convenient use duration.

Algorithm Contribution-Degree
BEGIN

```
n1: number of the number of processes
n2: number of the number of projects

'Step 1: computing contribution degree Knowledge-Project
For i= 1 to n2
   max← 0 ; min←0 ; Proc← 0
   For j=1 to n1
      If TabK-P[j] < MatP-R [i][j] Then
         min ← TabK-P[j]
      else
         min ← MatP-R [i][j]
      EndIf
      If min > max Then
         max ← min; proc← j
      EndIf
   EndFor
   'The retained path is the one passing through proc
   Res[i] ← max
   EndFor

'Step 2: Computing contribution degree Knowledge-Objective
max← 0 ; min← 0
For i=1to n2
   If TabR-O[i] < Res[i] Then
      min← TabR-O[i]
   Else
    min ← Res [i]
   EndIf
If min > max Then max ← min EndIf
EndFor
Contribution-degree-K-O ←max
```

END

Fig. 10. Contribution degree computing algorithm

Once all the decision rules are generated, the user may use the interface shown in Figure 12 to visualize the evaluation of each knowledge in respect to each criteria. Then, s/he should assign these knowledge into classes $Cl1$ or $Cl2$. Naturally, humain may provide some incoherent information when classifying these knowledge. To illustre this fact, consider knowledge $K7$ and $K8$ in the list shown in Figure 12 and suppose that $K7$ and $K8$ have the same evaluation in respect to all criteria. In this case, they should normally be assigned to the same class and not to different classe—as it is shown in Figure 12.

It is possible that the evaluation of a knowledge in respect to a given criterion is unavailable. This lack of information is designed by "?" symbol in the screen of Figure 12. This fact was one of many reasons for adopting DRSA instead of several other classification techniques as those based on outranking relation (e.g. Electre Tri), additive utility function (e.g. UTDAS) or hierarchical process discrimination (e.g. MHDIS) or other methods proposed in artificial intelligence where incoherences are eliminated before analysis. In fact, DRSA, which is based on rough set theory, is able to detect incoherence in the decision table, which are latter taken into account in the final decisions and not eliminated in early steps as the case with the above-cited techniques.

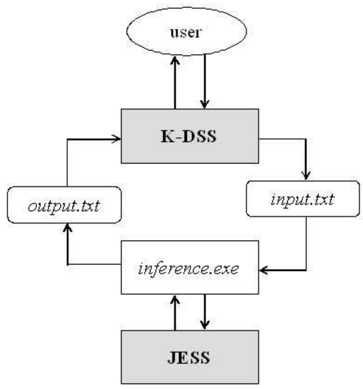


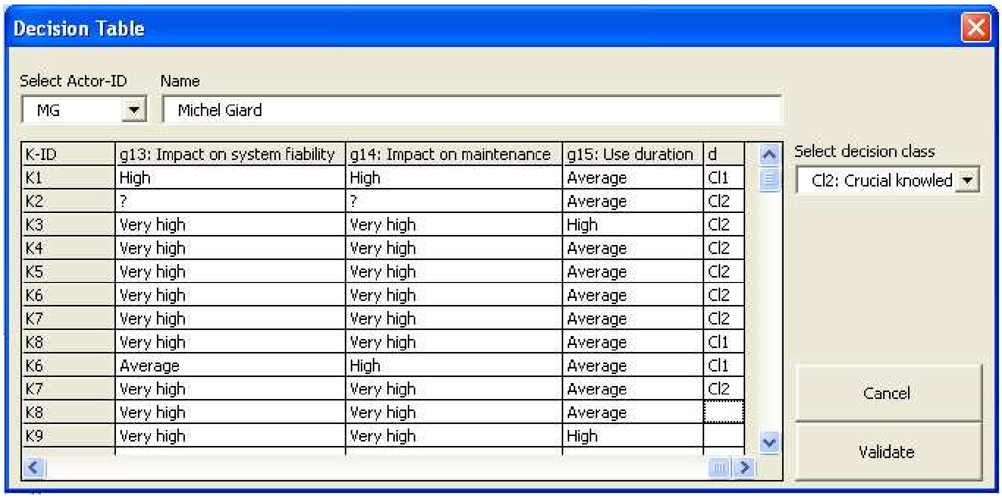
Fig. 11. K-DSS and JESS dialogue system

VI. CONCLUSION AND FUTURE WORKS

We have introduced K-DSS, a Decision Support System for identifying and evaluating crucial Knowledge. K-DSS is an implementation of a two phases-based methodology conducted and validated in the PSA Peugeot Citroën automobile company in France. Here, attention is especially devoted to present the conceptual and functional architectures of K-DSS. The implementation of K-DSS is also addressed.

In addition to reenforcing the capabilities of K-DSS, several points related to the methodology itself need to be investigated. Here, we mention two points. The first one concerns the need that the contribution degrees computing model take into account the natural (temporal) evolution of different projects concerned by the capitalization operation. For example, during our experiences at PSA Peugeot Citroën automobile company, some knowledge relative to the use of a chemical substance in the FAPz system were qualified as very important by the actors. Eight months later, this substance is no latter used in the project. One possible solution to tackle this problem is to use robustness analysis [21]. More precisely, this type of uncertainty may be modelled in terms of scenarios corresponding to the possible combinations of different values attributed by each actor to the contribution of each knowledge to each objective.

The second point is related to the first one and concerns the need to take into account imprecision and uncertainty at the database level. Fuzzy set seems to be a natural way to cope with this problem. This was partially shown in [2] where we have defined the class Knowledge as fuzzy concept. We have then associated to this class two extent properties [1], [5]: $P_{knowledge} = \{p_1, p_2\}$ where p_1 and p_2 are based on *level-of-tacit* and *degree-of-maturity* attributes, respectively—these two



K-ID	g13: Impact on system fiability	g14: Impact on maintenance	g15: Use duration	d
K1	High	High	Average	Cl1
K2	?	?	Average	Cl2
K3	Very high	Very high	High	Cl2
K4	Very high	Very high	Average	Cl2
K5	Very high	Very high	Average	Cl2
K6	Very high	Very high	Average	Cl2
K7	Very high	Very high	Average	Cl2
K8	Very high	Very high	Average	Cl1
K6	Average	High	Average	Cl1
K7	Very high	Very high	Average	Cl2
K8	Very high	Very high	Average	
K9	Very high	Very high	High	

Fig. 12. Decision table interface

attributes are not defined in the original model. By associating appropriate weighes w_1 and w_2 to the extent proprieties p_1 and p_2, the *degree of membership* of a piece of knowledge K_i to fuzzy class Knowledge may be computed as follows [1], [5]:

$$\mu_K(K_i) = \frac{\sum_{i=1}^{n} \rho_{P_K^i}(v_i) \cdot w_i}{\sum_{i=1}^{n} w_i},$$

where the number v_i is the value of the attribute of K_i on which the extent property p_i is defined and $\rho_{P_K^i}(.)$ represents the extent to which entity K_i verifies property p_i of fuzzy class K. The idea may easily be generalized to other classes of the model.

References

[1] R. Bouaziz, S. Chakhar, V. Mousseau, R. Sudah, and A. Telmoudi. Database design, implementation and querying within the fuzzy semantic model. *Information Sciences*, to appear.

[2] R. Bouaziz, S. Chakhar, and I. Saad. Membership functions definition in the fuzzy semantic model. In *Proc. of the International Conference: Sciences of Electronic, Technologies of Information and Telecommunications (SETIT'05)*, pages 266–275, Sousse, Tunisia, March 27-31 2005.

[3] B.Tseng and C. Huang. Capitalizing on knowledge: A novel approach to crucial knowledge determination. *IEEE Transactions on Systems, Man, and Cybernetics Part A: Systems and Humans*, 2005.

[4] J. G. Carbonell. *Derivational analogy: A theory of reconstructive problem solving and expertise acquisition*, chapter Reading in Knowledge Acquisition and Learning, pages 727–738. Morgan Kaufmann, 1986.

[5] S. Chakhar and A. Telmoudi. Extending database capabilities: Fuzzy semantic model. In *Proc. of the International Conference: Sciences of Electronic, Technologies of Information and Telecommunications (SETIT'04)*, pages 266–275, Sousse, Tunisia, March 15-20 2004.

[6] T.H. Davenport and L. Prusak. *Working knowledge: How organizations manage what they know*. Harvard Business School Press, 1998.

[7] R. Dieng, O.Corby, A.Giboin, and M. Rybière. Methods and tools for corporate knowledge management. Technical report, INRIA, 1998.

[8] S. Greco, B. Matarazzo, and R.Slowinski. Rough sets theory for multicriteria decision analysis. *European Journal of Operational Research*, 129(1):1–47, 2001.

[9] S. Greco, B. Matarazzo, R. Slowinski, and J. Stefanowski. An algorithm for induction of decision rules consistent with dominance principle. In W. Ziarko and Y. Yao, editors, *Proc. 2nd Int. Conference on Rough Sets and Current Trends in Computing*, pages 266–275, Banff, October 16-19 2000.

[10] M. Grundstein. *From capitalizing on company knowledge to knowledge management*, chapter 12, pages 261–287. The MIT Press, 2000.

[11] M. Grundstein, C. Rosenthal-Sabroux, and A. Pachulski. Reinforcing decision aid by capitalizing on company's knowledge. *European Journal of Operational Research*, 145(2):256–272, 2003.

[12] J.W. Grzymala-Busse. *Intelligent decision support. Handbook of applications and advances of the rough sets theory*, chapter LERSA system for learning from examples based on rough sets, pages 3–18. Kluwer Academic Publishers, Dordrecht, Boston, London, 1992.

[13] J.W. Grzymala-Busse. A new version of the rule induction system LERS. *Fundamenta Informaticae*, page 2739, 1997.

[14] G. Van Heijst, R. Van Der Speak, and E. Kruizinga. Organizing corporate memory. In *Knowledge acquisition for knowledge-based systems workshop*, pages 17–42, Banff, Canada, 1996.

[15] Ph. Lorino. *Méthodes et pratiques de la performance*. Editions d'Organisation, Paris, France, 2000.

[16] A. Macintosh. Corporate knowledge management: State-of-the-art review. International Symposium on the Management of Industrial and Corporate knowledge, 1994.

[17] Le Moigne. *La théorie du système générale, théorie de la modélisation*. P.U.F., 1977.

[18] J.B. Noh, K.C. Lee, J.K. Kim, J.K. Lee, and S.H. Kim. A case-based reasoning approach to cognitive map-driven tacit knowledge management. *Expert Systems with Application*, pages 249–259, 2000.

[19] I. Nonaka and H. Takeuchi. *The knowledge creating company*. Oxford University Press, 1995.

[20] Z. Pawlak. Rough sets. *International Journal of Information and Computer Sciences*, 11:341–356, 1982.

[21] B. Roy. A missing link in OR-DA: robustness analysis. *Foundations of Computing and Decision Sciences*, 23(3):141–160, 1998.

[22] I. Saad. *Une contribution méthodologique pour l'aide l'identification et l'évaluation des connaissances nécessitant une opération de capitalisation*. PhD thesis, University of Paris Dauphine, Paris, France, 2005.

[23] I. Saad, M. Grundstein, and C. Rosenthal-Sabroux. Locating the company's crucial knowledge to specify corporate memory: A case study in automotive company. In *Knowledge Management and Organisational Memory*, pages 75–84, Acapulco, Mexico, 2003. International Joint Conference on Artificial Intelligence.

[24] I. Saad, C. Rosenthal-Sabroux, and M. Grundstein. Improving the decision making process in the design project by capitalizing on company's crucial knowledge. *Group Decision and Negotiation*, 14(2):131–145, 2005.

[25] J. Stefanowski and D. Vanderpooten. *Rough sets, fuzzy sets and knowledge discovery*, chapter A general two stages approaches to rule induction from examples, pages 317–325. Springer, 1994.

[26] S. Tsuchiya. Improving knowledge creation ability through organizational learning. In *Proc. of ISMICK'93*, pages 27–28, UTC Compiègne, France, October 2003. International Joint Conference on Artificial Intelligence.