HAL
open science

Decision-Making Ontology for Information System Engineering

Elena Kornyshova, Rebecca Deneckere

To cite this version:

Elena Kornyshova, Rebecca Deneckere. Decision-Making Ontology for Information System Engineering. ER 2010 29th International Conference on Conceptual Modeling, Nov 2010, Vancouver, Canada. pp.104-117, 10.1007/978-3-642-16373-9_8 . hal-00662930

HAL Id: hal-00662930

https://paris1.hal.science/hal-00662930v1

Submitted on 24 May 2012

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Decision-Making Ontology for Information System Engineering

Elena Kornyshova, Rébecca Deneckère

CRI, University Paris 1 - Panthéon Sorbonne, 90, rue de Tolbiac,
75013 Paris, France
{elena.kornyshova,rebecca.deneckere}@univ-paris1.fr

Abstract. Information Systems (IS) engineering (ISE) processes contain steps where decisions must be made. Moreover, the growing role of IS in organizations involves requirements for ISE such as quality, cost and time. Considering these aspects implies that the number of researches dealing with decision-making (DM) in ISE increasingly grows. As DM becomes widespread in the ISE field, it is necessary to build a representation, shared between researchers and practitioners, of DM concepts and their relations with DM problems in ISE. In this paper, we present a DM ontology aimed at formalizing DM knowledge. Its goal is to enhance DM and to support DM activities in ISE. This ontology is illustrated within the requirements engineering field.

Keywords: Decision-making, Ontology, Information System Engineering.

1 Introduction

Information system (IS) conception, development, implementation, and every other process in IS engineering includes steps where several alternatives are considered and a decision must be made. Decision-making (DM) may be considered as an outcome of a cognitive process leading to the selection of an action among several alternatives. It might be regarded as a problem solving activity which is terminated when a satisfactory solution is found. With regard to IS engineering methodologies, the issue of DM has already been explored with respect to requirements engineering [1], to method engineering [2] [3], and, more generally, to systems engineering [4]. For instance, the GRL model allows to evaluate solutions according to their contribution to the goals [5]. Ruhe emphasized the importance of DM in SE along the whole life cycle [4]. Several examples of different DM methods application can also be mentioned: AHP for prioritizing requirements [6] and evolution scenarios [7]. Saeki uses weighting method to deal with software metrics [8]. Outranking and weighting methods are illustrated in the field of method engineering in order to select method fragments from a repository according to some project characteristics [3].

As shown in [4], engineering-related decisions may result from the need to satisfy practical constraints such as quality, cost or time. Ruhe stresses the importance of DM in the field of IS because of: (i) time, effort, quality and resources constraints; (ii) presence of multiple objectives; (iii) uncertain, incomplete and fuzzy information, and

(iv) complex decision space. However, the arguments to carry out final decisions are still poor, and choices are made in an intuitive and hazardous way [1] [4].

We consider the lack of DM in ISE at three levels: (i) at the *tool* level, (ii) at the *method* level, (iii) and at the *model* level. At the *tool* level, even if DM tools exist, there is none with a complete context-aware DM process. At the *method* level, intuitive and ad hoc decisions overshadow the method-based ones. At the *model* level, decisions are often ill-formulated. They are characterized, for instance, by poor understanding and description of decision problems, by misunderstanding of decision consequences, and by the lack of alternatives and criteria formalization.

We have developed the *MADISE* (*MAke Decisions in Information Systems Engineering*) approach to solve DM drawbacks at the method and model levels. The main goal of the MADISE approach is to guide IS engineers through DM activities. The MADISE approach includes three elements: DM ontology, MADISE process, and DM methodological repository. The *DM ontology* (DMO) is a representation of DM concepts for formalizing DM knowledge. The *MADISE process* is a generic DM process including main activities used for DM and explaining how to use DMO. The *DM methodological repository* provides a set of methodological guidelines for realizing DM activities.

The goal of this work is to present the DM ontology. Even if the number of researches dealing with DM in IS engineering increasingly grows [9], a complete DM ontology does not exist. DM becomes widespread in IS engineering field, so it is mandatory to build a shared representation of DM concepts and to show how these concepts are related to DM problems in IS engineering. The main goal of DMO is to represent concepts of the DM domain, as well as their properties and relations. The DM ontology fulfils the following needs:

- to clarify and organize DM concepts;
- to build a shared representation of DM concepts between researchers and practitioners;
- to show how these concepts are related to DM problems in IS engineering;
- to make DM knowledge reusable in similar IS engineering situations;
- to compare existing DM models in order to select an appropriate one;
- to validate the completeness of existing DM models;
- to support the creation of new DM models.

This paper is organized as follows. In Section 2, we explain the main concepts that we have used for building the DM ontology. We give an overview of the DM fundamentals and describe the DM ontology in Section 3. In Section 4, we validate DMO by applying it to a case from the requirements engineering field. We conclude this paper by presenting the possible applications of the DM ontology and our future works in Section 6.

2 Building a Decision-Making Ontology

In this section, we analyse different aspects that we have used for building DMO. We present a generic definition of the ontology concept; several classifications applied to DMO; DMO elements; DMO modelling way; and, finally, DMO goals.

Definition. The term ontology is taken from philosophy, where *Ontology* is a systematic account of *Existence*. The notion of "ontology" denotes the science of being and, with this, of descriptions for the organization, designation and categorization of existence [10]. Gruber was the first to formulate the term ontology in the field of Computer Science [11] and defined it as "an explicit specification of a conceptualization".

Gruber [11] has found the main principles for constructing ontologies in Computer Science and defined the main ontology elements, such as classes, relations, functions, or other objects. Since then, many approaches were developed for creating and applying ontologies [12] [13] [14] [15] [16] [17]. For instance, [10] defines the ontology applied to Computer Science as "the capture of the recognized and conceived in a knowledge domain for the purpose of their representation and communication". For [16], an ontology is a way of representing a common understanding of a domain. [12] considers ontology as a novel and distinct method for scientific theory formation and validation. However, all new definitions are based on the idea that Computer Science ontology is a way of representing concepts like in the Gruber's approach [16]. Ontologies could be linked to object models as the former express a formal, explicit specification of a shared conceptualization and the latter ones refer to the collection of concepts used to describe the generic characteristics of objects in object-oriented languages [13]. The main distinction between ontologies and object models relates to the semantic nature and shared conceptual representation of ontologies.

Classification. [16] presents several classifications of ontologies in Computer Science: by their level of generality (Guarino N., 1998 and Fensel D., 2004 classifications), by their use (Van Heijst G., Schereiber A.T. and Wieringa B.J., 1996 classification), and by the level of specification of relationships among the terms gathered on the ontology (Gómez-Perez A., Fernández-López M. and Corcho O., 2003 classification). According to these works, DMO can be defined as follows:

- according to the generality level: DMO is a domain ontology which captures the knowledge valid for ISE domain and describes the vocabulary related to the domain of DM in ISE;
- according to the use: DMO is a knowledge modeling ontology as it specifies the conceptualization of the DM knowledge;
- according to the specification level: DMO is a lightweight ontology, which includes concepts, concept taxonomies, relationships between concepts and properties describing concepts and which omits axioms and constraints.

Elements. DMO is a lightweight ontology including the following main elements: concepts, relations and properties [13] [16]:

- Concepts represent objects from the real world and reflecting the representational vocabulary from domain knowledge [11];
- Relations are relationships between concepts representing a type of interaction between concepts. Three types of relationships may be used in ontology: generalization, association, and aggregation [13];
- Properties (or attributes) are characteristics of concept describing its main particularities which are concise and relevant to the ontology's goals.

Modelling. [16] mentions several methods for modelling ontologies such as frames and first order logic, Description Logics, Entity-Relationship (ER) diagrams or Unified Modeling Language (UML) diagrams. UML is sufficient to model lightweight ontologies [16] and is a well-known modeling language. For this purpose, the UML formalism is already used, for instance, in an ontology for software metrics and indicators for cataloging web systems [18] or for representing domain ontologies and meta model ontology in requirements engineering [15]. For this reason, we have selected UML class diagram for representing DMO. In this case, each class represents a concept. Concept taxonomies are represented by generalization relationships. Relations between concepts are represented by association relationships. Concept properties are attributes of the corresponding classes.

Goals. In general, an abstract representation of phenomena expressed with a model must be relevant to the model's purpose [19]. Ontology is a conceptualization, which is an abstract, simplified view of the world that we wish to represent for some purpose [11]. This is the reason why goals for building DMO must be specified. We define in our approach the following goals of DMO based on the reviewed literature on ontologies in Computer Science [11] [12] [15] [16] [19] [20]:

1. Knowledge conceptualization. As ontologies represent concepts, they offer "ways to model phenomena of interest, and, in particular, model theories that are cast in the form of a conceptual framework in a much more rigorous fashion" [12]. In this manner, DMO provides a flexible way for conceptualizing DM knowledge.
2. Domain modelling. Ontology aims at modelling a specific domain [16]. [19] claims that "domain understanding is the key to successful system development". Domain understanding is usually represented via some form of domain modeling [19]. This allows representing complex real world objects within graphical and diagram representations understandable and accessible to experts and practitioners in diverse domains. Motivation for DMO is to have a unique model for DM knowledge.
3. Anchoring [19]. Ontology allows anchoring concepts – often abstract – to concrete application domains. From this view point, DMO aims at relating DM concepts to ISE.
4. Sharing representation. Agents must communicate about a given domain and have a common language within this domain. [11] calls this "ontological commitments". An ontology must include atomic concepts that any stakeholders, or other agents, can commonly have in a problem domain [15]. From this view-point, ontology is a compromise between different viewpoints, different stakeholders, or involved parties. [16] claims that "not only people, but also applications must share a common vocabulary, that is, a consensus about the meaning of things". This consensus is reached by building ontologies, which are one of the solutions for representing this common understanding. Therefore, all participants of DM process (such as IS engineers, method engineers, users, and stakeholders) and also applications must share a common understanding of a DM problem.
5. Model validation. In Software Engineering, a specific ontology could be taken as reference point to validate a model that acts over a particular domain [16]. DMO application enables validating existing DM models or new ones in the ISE

domain. The following criteria may be tested: consistency, completeness, conciseness, expandability, sensitiveness [20].

3 DMO: Decision-Making Ontology

In this section, we describe the DM ontology. After an introduction in DM fundamentals, we present DMO and, then, the organization of DM knowledge into DM method components.

3.1 Decision-Making Fundamentals

A decision is an act of intellectual effort initiated for satisfying a purpose and allowing a judgement about the potential actions set in order to prescribe a final action. Bernard Roy defines three basic concepts that play a fundamental role in analysing and structuring decisions [21]: decision problem, alternatives (potential actions), and criteria.

The decision *problem* [21] can be characterised by the result expected from a DM. When the result consists in a subset of potential alternatives (most often only one alternative) then it is a *choice problem*. When the result represents the potential alternative affectation to some predefined clusters, then it is a *classification problem*. When the result consists in a potential collection of ordered alternatives then it is a *ranking problem*.

The concept of *alternative* designates the decision object. Any decision involves at least two alternatives that must be well identified.

A *criterion* can be any type of information that enables the evaluation of alternatives and their comparison. There are many different kinds of criteria: intrinsic characteristics of artefacts or processes, stakeholders' opinions, potential consequences of alternatives etc. When dealing with criteria, the engineer must determine "preference rules", i.e. the wishful value of criterion (for example, max. or min. for numeric criterion) according to a given need.

Herbert Simon (1978 Nobel Prize in Economics) was the first to formalize the decision-making process. He suggested a model including three main phases: intelligence, design, and choice (I.D.C. model) [22]. Intelligence deals with investigating an environment for conditions that call for decisions. Design concerns inventing, developing, and analyzing possible decision alternatives. Choice calls for selecting an alternative from possible ones.

This process was modified and extended in different ways. Currently, the commonly agreed and used decision-making steps are defined as follows [23]:

- define problem (necessity to define priorities),
- identify problem parameters (for instance, alternatives and criteria),
- establish evaluation matrix (estimate alternatives according to all criteria),
- select method for decision making,
- aggregate evaluations (provide a final aggregated evaluation allowing decision).

These are the basic notions of DM. However, an additional analysis of the DM literature is required in order to build a complete DM ontology. We have used the following references for completing the DM knowledge: [24] [25] [26] [27] [28] [29].

3.2 Decision-Making Concepts Ontology

Based on the State-of-the-Art and DM knowledge we have developed the DM ontology which is a domain knowledge lightweight ontology including concepts, attributes and relationships. This ontology represents DM knowledge as a UML class diagram (See Fig. 1).

In the following, we describe the DM ontology and give some additional explanations within a common example, which is a project portfolio management (PPM), for instance a project of an ERP purchase. A more detailed description of the DM concepts ontology is given in the Appendix. The concepts, attributes, and relationships are respectively shown in Tables 1, 2, 3.

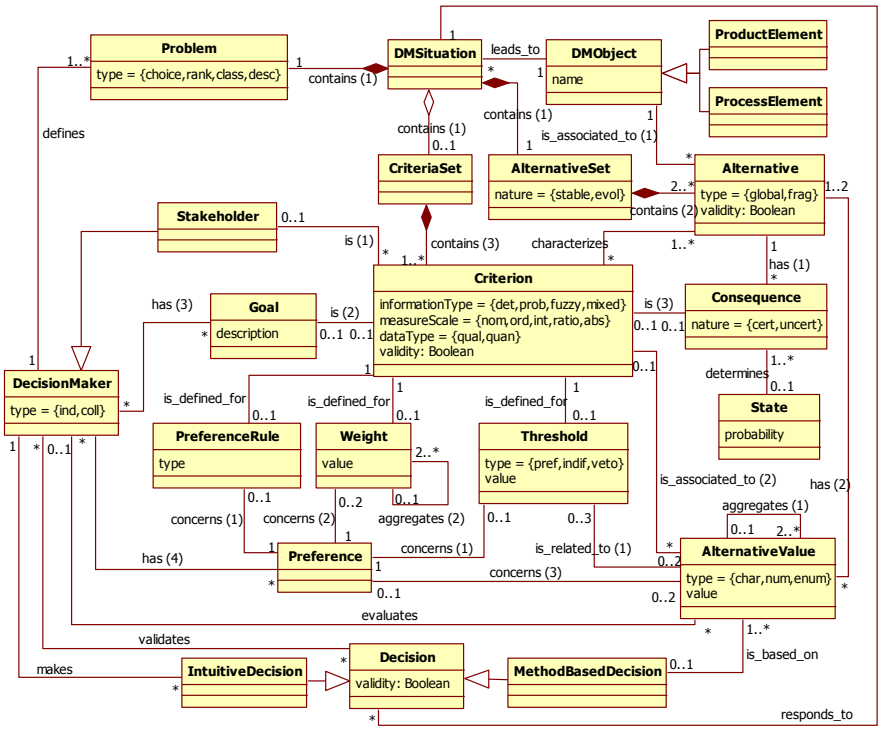

Fig. 1. Decision-Making Ontology.

The starting point for analyzing the DM concepts ontology is the *DM situation*. The DM situation is an abstract concept which puts together the main DM elements and

describes a concrete case of DM dealing with a given *DM object*. The DM object can be a *process* or a *product element*. In the case of PPM, the `project` is the DM object, which is a `product element`.

A given DM situation contains a *DM problem* and a *set of alternatives*. In PPM, the problem is a choice of one or more relevant projects; and alternatives can be `SAP ERP` project, `Oracle e-Business Suite` project, and `OpenERP`. The DM situation can also contain a *set of criteria*. It is not mandatory as a decision could be made without analyzing criteria. Criteria can have different nature. It can be: (i) intrinsic characteristics of alternatives (*characterizes* relationship), (ii) future consequences of alternatives depending on the future state (*is(3)* relationship), (iii) decision-makers' goals (*is(1)* relationship), or (iv) decision-makers themselves (in this case, they have a role of *stakeholders* according to the *is(2)* relationship). Regarding to the PPM case, two criteria can be considered: `purchase cost` and `maintenance cost`. The first one is known directly and constitutes a `characteristic` of a project. The second one depends on several factors (for instance, project duration) and implies the `consequences` of the ERP implementation in the future.

The criteria could be analyzed in order to know their *weights*, *preference rules*, and *thresholds*. In our case, weights could be `equal` (same importance of two criteria); the preference rule is the `minimization` of two costs; a threshold could be established in order to indicate the `maximal acceptable cost` of the ERP purchase and maintenance.

All alternatives are evaluated according to the identified criteria in order to obtain the *alternative values*. In the ERP purchase case, a value matrix (3 X 2) will be constituted. The alternative values can be aggregated in order to produce a unique value by alternative (*aggregates* relationship). For instance, it can be a weighted sum. In the PPM case, the two ERP costs will be `added` for each alternative ERP. Thus, each ERP will have only one value allowing to compare all ERP in an easier way. Based on these aggregated values, a `method-based decision` will be made.

Decision-makers can participate in DM by several ways. They *define* the DM problem; *have goals* and *preferences* with regard to preference rules, weights and thresholds. They can also become criteria as a particular decision-maker type – *stakeholder*. Decision-makers evaluate *alternatives*, *validate* decisions and *make intuitive decisions*. In our case, decision-makers participate in the `definition` of the DM problem, in the `establishment` of weights, preference rules, and thresholds. They also `validate` the final decision.

Both method-based and intuitive decisions *are related to* the DM situation. Each DM situation can lead to either none or several decisions.

3.3 Decision-Making Method Components

DMO elements are organized into DM method components in order to make easy their use. The notion of method component is inspired from the Method Engineering domain [30]. The DM component model is shown on Fig. 2. It includes six concepts: component, actor, intention, concept, activity, and context.

Component. A DM method component is a reusable building bloc of a DM method that can be used separately. Each DM method component may contain several method components, which, in turn, may also be decomposed in other more simple components.

Actor. Actors participating in DM can have three main roles: stakeholder, IS engineer, and DM staff. A *Stakeholder* defines the decision problem, sets goals, expresses preferences on alternatives and criteria [23] and validates the final decision. An *IS engineer* evaluates alternatives and makes a proposal for DM to stakeholders. *DM staff* is responsible for assisting stakeholders and IS engineers in all stages of the DM process [23]. DM staff includes a machine support for DM. In this case, this is a system actor. If actors are human, we call them decision-makers. Decision-makers have the same roles as actors but have a complementary property, which is their type: individual or collective. A collective decision-maker is a group of decision-makers having the same goals and preferences and acting as a unique actor. Actors contribute to the DM process at different stages. It is obvious that the same actor can play different roles in a specific DM process.

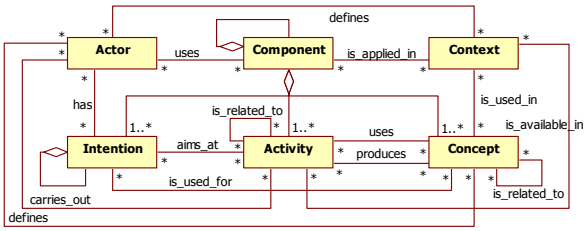

Fig. 2. DM Method Component.

Intention. This concept describes goals that actors have within the DM process. It is represented as a taxonomy. The main intention is to find a solution of a problem having a DM nature. This intention is decomposed into more detailed ones, for instance, define alternatives, define relative importance of criteria, and so on.

Concept. Concepts are objects used in DM, for instance, the DM problem (choice, ranking, classification), alternative, criterion, and so on. The DM concepts and their relationships are complex. We have organized them into an ontology of DM concepts, which is described is details in the following sub-section.

Activity. The activity concept describes elementary actions used for making decisions. The activity taxonomy contains activities (enumerate alternatives, calculate a weighted sum, calculate an aggregate value etc.) and the possible relationships between them: composition, precedence and parallel execution.

Context. The context describes conditions in which decisions are made. The context is represented as a taxonomy of characteristics, such as cost, time, etc.

These meta-concepts are related to each other as follows. An actor *has* the intention to make a decision in order to resolve a problem. He *defines* several DM concepts and *carries out* different DM activities. DM activities *use* various concepts and *produce* other ones. Both DM concepts and activities are related to intentions in order to show for which reason they are used in a DM process (*is_used_for* and *aims_at* relationships).

They are also related to a context in order to indicate the conditions in which they can be used (*is_used_in* and *is_available_in* relationships). A combination of intentions, concepts and activities (composition link between component and intention, concept, and activity) represents a component, as each component contains methodological information about its application. Components are related to a context in order to indicate context characteristics, in which they can be applied (*is_applied_in* relationship). The context is defined by involved actors (*defines* relationship).

4 An Application Case: the REDEPEND Approach

This section aims at validating the DM ontology. Our goal is to show how existing DM models could be expressed through the DM ontology. We have chosen an existing and well known DM method dealing with requirements engineering: the REDEPEND approach [6]. We capture its DM model and express it through DMO, i.e. DMO concepts, attributes and relationships are used in order to represent the REDEPEND approach.

The REDEPEND approach uses requirements for selecting candidate tasks, which represent possible alternatives. It is based on the AHP (Analytic Hierarchy Process) DM method. The AHP, proposed by T.L. Saaty [31], includes the pair-wise comparison between alternatives and/or criteria and the aggregation of the comparison results into a quantitative indicator (score). The REDEPEND approach integrates the AHP and *i**, which is a well-known requirements modeling formalism.

Fig. 3 represents the DM concepts used in the REDEPEND approach. The DM situation in the REDEPEND approach is characterized as follows. The DM problem is *ranking*. The *DM object* is a task, which can be a scenario (*process element*) or a goal (*product element*). Tasks represent *alternatives*, which are *fragmented*, i.e. they can be dependent one another. All alternatives are *true* as the REDEPEND approach does not contain a module for validating them. The alternative set is *evolving* as it can change through time. One or more decision-makers (*individual stakeholders*) define goals and soft goals. Goals and soft goals represent requirements and are considered as *criteria* in the given model. These goals are *determinist*; their measure scale is *nominative*; the data type is *qualitative*; and they are *valid*.

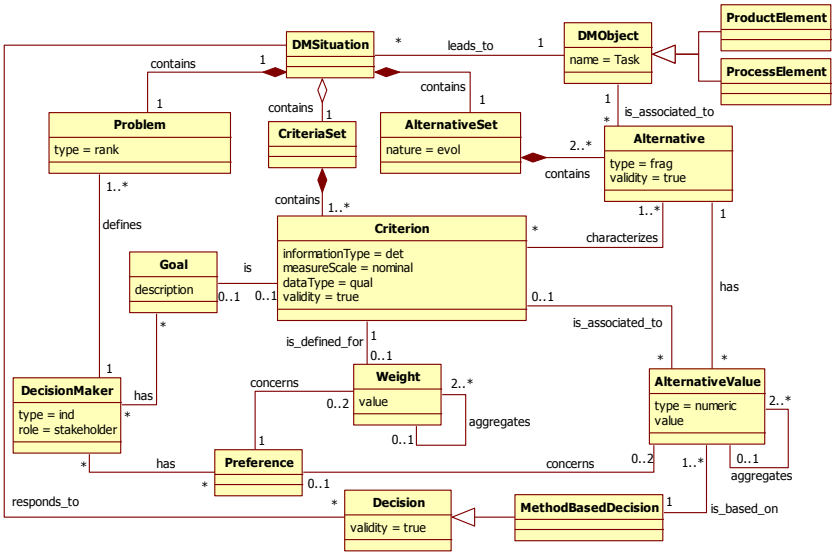

Fig. 3. DMO: Application to the REDEPEND Approach.

Decision-makers make pair-wise comparisons of tasks and goals. In this way, they express preferences on weights and alternatives values (*concerns* relationships). For instance, they compare each pair of alternatives according to a criterion and give a *numeric value* to it. A value can vary from 1 (equal importance) to 9 (absolute importance) in accordance with the basic AHP method. This constitutes the elementary *alternative value*. These values for all alternatives are then aggregated in order to rank alternative against the given criterion (*aggregates* relationship). The same analysis is made between alternatives for each criterion and between criteria in order to prioritize them too (class *weight* and relationship *aggregates* respectively). The ranked alternatives and criteria are computed for the final alternatives' ranking in order to make decision (*is_based_on* relationship).

Fig. 4. illustrates DM method components used in the REDEPEND approach. REDEPEND implements the AHP method and has three related components (See Fig. 4.A), which are used for the pair-wise comparison of tasks and pair-wise comparison of criteria (two instances of *PWComponent*), and for the computation of candidates ranking (an instance of *CompComponent*).

We detail the PWComponent dealing with alternatives ranking in Fig. 4.B. The PWComponent contains the intention which is to *prioritize candidates*. This component includes alternative values as concepts and two activities. The first activity *normalizing values* uses alternative values for producing normalized alternative values. The second activity *calculating relative values* (used for calculating relative rating of each task) uses normalized alternative values for producing ranked alternative values.

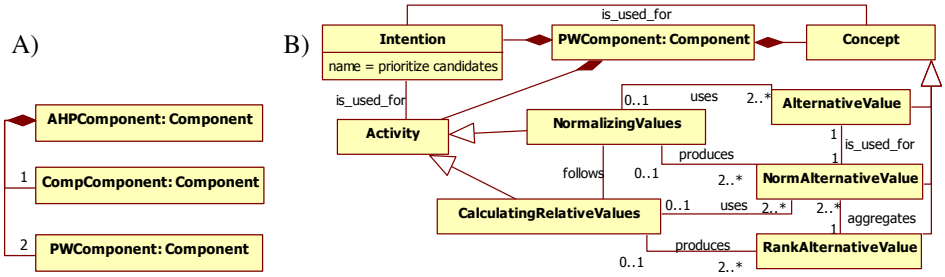

Fig. 4. A) REDEPEND DM Method Components; B) PWComponent of REDEPEND Approach.

5 Conclusion

In this paper, we have presented the DM ontology. DMO aims at representing and formalizing DM knowledge. It includes concepts, their properties and relationships organized into two levels. We have validated DMO by applying it to a well-known DM method from the requirements engineering field.

The main goal of DMO is to enhance and facilitate DM. It supports IS engineers in DM activities. Therefore, IS engineers could use this ontology in every case of DM. This implies that DMO includes all necessary elements and links for supporting DM in various situations.

We foresee different applications of DMO as follows. Firstly, DMO helps to overcome the abovementioned drawbacks of DM in IS engineering at the model and method levels. For instance, at the model level, it contributes to a better comprehension of the DM problem and decision consequences; it allows formulating DM situations in a clear way shared between different DM actors. At the method level, DMO encourages the usage of DM scientific methods for decisions. In fact, by showing how IS engineering concepts are related to DM ones, DMO makes the usage of different methods and models from the operational research field easier. Secondly, DMO responds to practical needs such as the validation of existing DM methods and models, their possible enhancing by adding different DM components, and the assistance in the creation of new ones.

As the definition of this ontology was motivated by the necessity to support the generic DM process MADISE, our future research includes (i) validation of the MADISE approach and (ii) development of the DM methodological repository.

6 References

1. Ngo-The, A. and Ruhe, G. Decision Support in Requirements Engineering, In Engineering and Managing Software Requirements, Ed. By A. Aurum and C. Wohlin, 267-286 (2005)
2. Aydin, M.N. Decision-making support for method adaptation, Ed. Enschede, Netherlands (2006)

3. Kornyshova, E., Deneckère, R., and Salinesi, C. Method Chunks Selection by Multicriteria Techniques: an Extension of the Assembly-based Approach, Situational Method Engineering (ME), Geneva, Switzerland (2007)
4. Ruhe, G. Software Engineering Decision Support – Methodology and Applications. In: Innovations in Decision Support Systems (Ed. by Tonfoni and Jain), International Series on Advanced Intelligence, Volume 3, 143-174 (2003)
5. Amyot D., Mussbacher G., URN: Towards a New Standard for the Visual Description of Requirements, In proceeding of 3rd Int. WS on Telecommunications and beyond: the broader applicability of SDL and MSC (2002)
6. Maiden, N.A.M., Pavan, P., Gizikis, A., Clause, O., Kim, H., and Zhu X, Integrating Decision-Making Techniques into Requirements Engineering, REFSQ'02, Germany (2002)
7. Papadacci, E., Salinesi, C., and Sidler, L. Panorama des approches d'arbitrage dans le contexte de l'urbanisation du SI, Etat de l'art et mise en perspective des approches issues du monde de l'ingénierie des exigences. special issue ISI Journal (2005)
8. Saeki, M. Embedding Metrics into Information Systems Development Methods: An Application of Method Engineering Technique. Proceedings of the CAISE'03, Klagenfurt/Velden, Austria, 374-389 (2003)
9. Kou, Gang, Peng, Yi, A Bibliography Analysis of Multi-Criteria Decision Making in Computer Science (1989-2009), In Cutting-Edge Research Topics on Multiple Criteria Decision Making, Communications in Computer and Information Science, Volume 35. Springer Berlin Heidelberg, 68-71 (2009)
10. Rebstock M., Fengel J. and Paulheim H., Ontologies-Based Business Integration, Springer (2008)
11. Gruber Thomas R., Toward Principles for the Design of Ontologies Used for Knowledge Sharing, In International Journal Human-Computer Studies 43, p.907-928. (1993)
12. Akkermans H. and Gordijn J., Ontology Engineering, Scientific Method and the Research Agenda, in S. Staab and V. Svatek (Eds.), International Conference on Knowledge Engineering and Knowledge Management No15 EKAW 2006 (4248), Lecture Notes in Computer Science Springer, Tcheque Republique, 112-125 (2006)
13. Batanov D. N., Vongdoiwang W., Using Ontologies to Create Object Model for Object-Oriented Software Engineering, In Ontologies: A Handbook of Principles, Concepts and Applications in Information Systems, Sharman R., Kishore R. and Ramesh R., Springer US, 461-487 (2007)
14. Hevner, A., March, S., Park, J., Ram, S., Design science research in information systems. MIS Quarterly 28(1), 75–105 (2004)
15. Kaiya H., Saeki M., Using Domain Ontology as Domain Knowledge for Requirements Elicitation, , 14th RE Conference Volume 11-15, 189 – 198 (2006)
16. Sánchez D. M., Cavero J. M. and Martínez E. M., The Road Toward Ontologies, In Ontologies: A Handbook of Principles, Concepts and Applications in Information Systems, Sharman R., Kishore R. and Ramesh R., Springer US, 3-20 (2007)
17. Wieringa, R., Maiden, N., Mead, N., Rolland, C.: Requirements engineering paper classification and evaluation criteria: A proposal and a discussion. Requirements Engineering 11(1), 102–107 (2006)
18. Martín M., Olsina L., Towards an Ontology for Software Metrics and Indicators as the Foundation for a Cataloging Web System, In Proceedings of the First Conference on Latin American Web Congress Washington, DC, USA, 103-113 (2003)
19. Offen R., Domain understanding is the key to successful system development, Requirements engineering, vol. 7, no3, Springer, London, Royaume-Uni (2002)
20. Gómez-Pérez A., Evaluation of ontologies, Workshop on Verification and Validation at DEXA, Vienna, Autriche, vol. 16, no 3 , (2001)
21. Roy, B. Paradigms and challenges, Book chapter, In Multiple Criteria Decision Analysis - State of the Art Survey, Springer. editor(s) J. Figueira, S. Greco, M. Ehrgott, (2005) 3-24

22. Simon, H.: The New Science of Management Decision, Harper&Row (1960)
23. Guidebook to decision-making methods, Developed for the Department of Energy, by Dennis Baker, Donald Bridges, Regina Hunter, Gregory Johnson, Joseph Krupa, James Murphy, Ken Sorenson, (2001)
24. Bouyssous D., Marchant Th., Pirlot M., Perny P., Tsoukias A., Vincke Ph., Evaluation and decision models: a critical perspective, Kluwer Academic Publishers, USA (2000)
25. Hanne T., Meta Decision Problems in Multiple Criteria Decision Making. Chapter 6 in T. Gal, T.J. Stewart, T. Hanne (ed) Multiple Criteria Decison Making-Advances in MCDM Models, Algorithms, Theory and Applications. International Series in Operations Research and Management Science, Volume 21. Springer, Kluwer (1999)
26. Keeney R.L. and Raiffa H., Decisions with Multiple Objectives: Preferences and Value Trade-Offs, Cambridge University Press (1993)
27. Bouyssou D., Outranking methods, In Encyclopedia of Optimization, Kluwer (2001)
28. Ballestero E., Romero C., Multiple criteria decision making and its applications to economic problems, Kluwer Academic Publishers, Netherlands (1998)
29. Vincke, P., L'aide multicritère à la decision (In French), Edition Ellipses, Edition de l'Université de Bruxelles, Bruxelles, Belgique (1989)
30. Ralyté, J., Deneckere, R., Rolland, C.: Towards a Generic Model for Situational Method Engineering, in proceedings of the CAISE'03, Springer Verlag, Velden, Austria (2003)
31. Saaty T.L., The Analytic Hierarchy Process, NY, McGraw Hill (1980)

Appendix: DMO Description

Table 1. Decision-Making Ontology: Glossary of concepts.

Concept Name	Description
Alternative	Possible action available for decision-making.
AlternativeSet	Set of alternatives available in a given DM situation.
AlternativeValue	Evaluation of an alternative according to a criterion or an aggregated alternative value.
Consequence	Impact that an alternative can have whether it is realized following the decision made.
CriteriaSet	Set of criteria available in a given DM situation.
Criterion	Information of any kind that enables the evaluation of alternatives and their comparison.
Decision	Act of intellectual effort initiated for satisfying a purpose and allowing a judgment about the potential actions set in order to prescribe a final action. It can be an IntuitiveDecision or MethodBasedDecision.
DecisionMaker	Actor contributing to the DM process at its different stages.
DMObject	Artifact being the subject of decision-making (ProductElement of ProcessElement) in IS engineering.
DMSituation	Set of specific conditions of DM dealing with a given DM object.
Goal	Intention or a projected state that a decision-maker intends to achieve.
IntuitiveDecision	Decision made 'on the fly' without using a DM method.
MethodBasedDecision	Decision based on the application of different DM methods.
Preference	Preference that a decision-maker have on different DM situation elements of alternatives and criteria.
PreferenceRule	Wishful value of a criterion according to a given need.
Problem	Result expected from a DM
ProcessElement	DM object corresponding to a process in IS engineering.
ProductElement	DM object corresponding to a product in IS engineering.
Stakeholder	Particular role of a decision-maker, which defines the DM problem, sets goals,

	expresses preferences on alternatives and criteria and validates the final decision.
State	State of the environment affecting alternative consequences in the future.
Threshold	Acceptable value for alternative values expressed for a given criterion.
Weight	Relative importance of a criterion.

Table 2. Decision-Making Ontology: Attributes Description.

Concept	Attribute	Description and/or Domain
Alternative	type	Type of alternative: global or fragmented.
	validity	Validity of an alternative, which is a Boolean value.
AlternativeSet	nature	Nature of the alternative set, which is stable or evolving.
AlternativeValue	type	Type of an alternative value: character, numeric, or enumeration.
	value	Value of an alternative.
Consequence	nature	Nature of a consequence, which can be certain or uncertain.
Criterion	information-Type	Type of information on a criterion: determinist, probabilistic, fuzzy, or mixed.
	measureScale	Measure scale: nominal, ordinal, interval, ratio, and absolute.
	dataType	Type of data: qualitative or quantitative.
	validity	Validity of a criterion, which is a Boolean value.
Decision	validity	Validity of a decision, which is a Boolean value.
DecisionMaker	type	Type of a decision-maker, which can be individual or collective (a group of decision-makers having the same goals and preferences and acting as a unique decision-maker).
DMObject	name	Name of a DM object.
Goal	description	Description of a goal.
PreferenceRule	type	Type of preference, for instance, a function (max, min), or an ordered list.
Problem	type	Problem type, which can be a choice, a ranking, a classification,.
State	probability	Probability of a state realization in the future.
Threshold	type	Threshold type: preference, indifference, or veto thresholds.
	value	Numeric value of a threshold.
Weight	value	Numeric value of a weight.

Table 3. Decision-Making Ontology: Relationships Description.

Relationship Name	Description
aggregates (1)	An alternative value can be an aggregation of at least two values which describe this alternative according to different criteria.
aggregates (2)	A weight value can be an aggregation of at least two values.
characterizes	Each criterion characterizes one or more alternatives. Each alternative can be characterized by one or more criteria.
concerns (1)	A preference may concern a threshold or a preference rule.
concerns (2)	A preference may concern a weight or two weights in the case of pair-wise comparisons.
concerns (3)	A preference may concern an alternative value or two alternative values in the case of pair-wise comparisons. An alternative value may be defined by a preference of a decision-maker.
contains (1)	Each DM situation contains a problem and an alternative set and can contain a criteria set.
contains (2)	An alternative set contains a least two alternatives.
contains (3)	A criteria set contains one or more criteria.
defines	A decision-maker defines a problem for each DM situation. He (she) can define several problems for different DM situations.
determines	A state can determine one or more consequences.
evaluates	A decision-maker can evaluate one or more alternatives.
has (1)	An alternative can have one or more consequences; each consequence is related to an alternative.

has (2)	An alternative can have several values; each alternative value is related to one or two alternatives.
has (3)	A decision-maker can have several goals. The same goal may be shared by several decision-makers.
has (4)	A decision-maker can have several preferences. The same preference may be shared by several decision-makers.
is (1)	A stakeholder can be a criterion in one or more DM situations.
is (2)	A goal can be a criterion in a given DM situation.
is (3)	A consequence can be a criterion in a given DM situation.
is_associated_to (1)	None or several alternatives are associated to a DM object.
is_associated_to (2)	None or several alternative values can be associated to a criterion.
is_based_on	A method-based decision is based on one or more alternative values.
is_described_by	Each DM situation is described by none or several DM characteristics.
is_defined_for	A preference rule, a threshold, or a weight can be defined for a criterion.
is_related_to (1)	A threshold can be related to one or two alternative values.
leads_to	Each DM situation leads to a DM object. A DM object can be related to several DM situations.
makes	A Decision-maker can make intuitive decisions. An intuitive decision is made by a decision-maker.
responds_to	A decision responds to a DM situation. A DM situation can be related to none or several decisions.
validates	A decision-maker can validate decisions; a decision can be validated by none or several decision-makers.