A framework for organizing the space of decision problems with application to solving subjective, context-dependent problems

Yoram Reich*, Adi Kapeliuk

School of Mechanical Engineering, Faculty of Engineering, Tel Aviv University, Tel Aviv 69978, Israel

Received 25 October 2003; received in revised form 6 May 2004; accepted 6 May 2004
Available online 11 June 2004

Abstract

Developing Decision Support Systems (DSS) is a difficult task, prone to errors and susceptible to colossal failures. In order to alleviate this difficulty and its consequences, we propose a framework for organizing the space of decision problems based on problem characteristics. This organization classifies decision problems into classes that exhibit intra-class similarity. We show how classifying a new decision problem into one of the existing classes provides assistance in designing the DSS for the new problem. The classification could be based simply on problem characteristics or on the structure of the problem–solution relationships. We further show that a DSS whose goal is to support the development of DSS can be developed with the assistance of related DSS.

The framework we propose could best serve researchers and practitioners if the reporting of DSS projects would follow the structure presented in this paper. In this way, additional data would be accumulated and would be available for refining the classification of the space of DSS. This in turn, would improve the support for building new DSS.

© 2004 Elsevier B.V. All rights reserved.

Keywords: Case-based reasoning; Software development; Influence knowledge graph; Decision problems characteristics; Clustering

1. Introduction

The majority of information technology projects fail to achieve some of their stated goals [8,37]. This is the fate of simple or complex projects such as executive information system implementations [27]. Some failures could be financially colossal [35]. Decision support systems (DSS) are among the most complex products as they intervene in, and modify, the core of work practices. According to common definitions, DSS encompass a wide variety of interactive, flexible, adaptable computer-based systems that help decision makers utilize data and models to solve unstructured or semi-structured problems [13,19,39]. We refer to the set of all DSS as the space of DSS. In view of the issues cited elsewhere as contributing to project failures [8,27,37,41], DSS susceptibility to failure is high.
The broad definition of DSS makes them applicable solutions for addressing diverse and heterogeneous problems, referred to as the space of decision problems. Indeed, the area of DSS has grown rapidly in recent years; new technologies and tools have been developed and applications implemented [2,38]. The abundance of technologies, tools, and development methods, do not ease developers’ tasks. In contrast, it becomes extremely difficult to determine which technology and development method is suitable to a particular problem context. In order to alleviate developers’ difficulties, we propose a framework that allows organizing the space of decision problems and systematically finding an appropriate development plan for addressing a given problem. By using previous successful experiences, the risks in developing DSS could be alleviated. The framework is quite simple, making it useable to a wide range of users. The motivation for this framework is further elaborated in this section and its details are developed in subsequent sections.

Four obvious dimensions for organizing the space of problems are (Fig. 1): the task to be solved, available technologies, organizations, and the people involved [22]. If we take the context to be the interaction between these dimensions with the addition of factors external to them (e.g., competitors, markets), we arrive at the evolving situation in (c) [32]. This model is reminiscent of Holsapple and Wenhong’s [14] framework for studying organizational computing and to Ariav and Ginzberg’s [1] systemic view of DSS.

An early classification of decision types was structured vs. unstructured [19]. Another classification of problems or tasks to be addressed by DSS is based mostly on broad classes of problems [18]: planning, design, diagnosis, monitoring, maintenance, etc. Similarly, technologies and approaches available for implementing DSS are sub-divided into several types that evolve continually including [38]: group decision support systems (GDSS), expert systems (ES), intelligent agents, artificial neural networks (ANN), executive information systems (EIS), knowledge management systems (KMS), and hybrid support systems. Another classification of DSS suggests the following classes of solutions: [15]: text-oriented DSS, database-oriented DSS, spreadsheet-oriented DSS, solver-oriented DSS, rule-oriented DSS, and compound DSS.

The organizations and their people vary significantly. Organizations have different strategic goals and value systems, structures, and practices. Their structure and management styles also differ. The people involved may also come from different cultures and value systems that affect their working and communication styles.

Fig. 1. Issues affecting DSS implementation ((a) and (b) were adapted from Ref. [22]).
Finally, it has long been recognized that there is no single development method for information systems [23]. Rather, there are numerous implementation processes available for DSS development including [4]: ad hoc, waterfall, iterative development, prototyping, exploratory, spiral, and reuse model. In fact, Avison and Fitzgerald [2] discuss 25 methodologies, 29 techniques, and 8 tools for information systems development. The most effective implementation process should be based on the implementation context and its expected evolution. Avison and Taylor [3] associated development methods with DSS problem classes. They concluded that most problems are complex DSS requiring a combination of multiple development methods. A similar observation has led to exploring different contingency models that integrate several development methods for designing information systems [42]. It turns out that there could be many such integration methods. Consequently, among the unanswered questions regarding development methods are [42]: how to select contingency models, what is the influence of culture on the choice of appropriate model, and how rigor is maintained in light of the flexibility in choosing methods?

We can summarize by asserting that the space of DSS is combinatorial and presently elusive: DSS literature does not offer a framework for understanding and managing this space. This echoes Ariav and Ginzberg’s [1] statement that “there has been no effective integration of these diverse elements to date, and this lack of a unified approach to the subject has hampered efforts to develop a solid basis for DSS design.” Ariav and Ginzberg offered a systemic view that provided a basis for moving towards a better understanding of DSS. They offered a taxonomy of factors and discussed the consequences of different values of these factors (e.g., lack of structure leads to high level of user interaction). Pearson and Shim [26] created five DSS classes from an empirical questionnaires study regarding DSS capabilities. They characterized those classes using 10 of the environmental and role factors identified by Ariav and Ginzberg [1]. As a result, they offered a six-step process of DSS development. Nevertheless, the design step received minimal support.

We believe that there is a way for creating a better and usable organization of decision problems and providing it is made simple and useable, could be beneficial to DSS developers. The feasibility of this endeavor can only be empirically validated. In this paper, we make a further step towards creating such a framework. Our goal is to show that the following claims hold: (1) it is possible to characterize decision problems (or structure the space of decision problems), (2) such a characterization could indeed create meaningful problem classes, and (3) these problem classes could be addressed by corresponding DSS technologies and development processes. While the first and second claims are similar to Pearson and Shim’s [26] hypotheses, our characteristics are more detailed and extendable. They are driven by detailed problem properties rather than general survey questions. We expect the classification to be dynamic as additional decision problems are added to the analysis. We also expect that such a classification would be context-dependent, i.e., it would depend on the particular experience of the organization(s) or community using it. Finally, the third claim provides concrete assistance with the development of DSS that is one of the most critical issues in DSS research and practice [25]. More specifically, the contribution of the paper is multifaceted, as now discussed.

First, at a concrete level, we present a class of complex problems that are quite frequent in practice yet as a class received little or no support thus far (see Appendices A–D). We characterize this class of problems and develop in a systematic manner a solution for one problem: a school dropout prevention DSS. The systematic solution exposes the rationale underlying the development process. We tested this system in lab settings and found it promising for improving the work of attendance officers who are in-charge of school dropout prevention. We contend that similar solution architecture will work also for other similar problems and illustrate this through an example.

Second, at a general level, we propose that the set of characteristics used for describing the class of decision problems is useful for characterizing decision problems in general and for differentiating between different classes of decision problems. Further, we propose that problems in these classes could be addressed by similar solutions. The first contribution supports this proposition.

The remainder of this paper is organized as follows. Section 2 details three hypotheses and presents a set of characteristics of decision problems. Section 3 discusses the sufficiency of these characteristics. We
show how knowledge about decision problems can be collected through a knowledge acquisition tool. An analysis of a short knowledge acquisition session illustrates the first and second hypotheses. Section 4 describes the development of a DSS for one problem and Section 5 illustrates its reuse in one related problem. This provides support for the third hypothesis. Section 6 discusses the consequences of this study to building a DSS for assisting the planning of DSS programs and Section 7 concludes the paper.

2. Characterizing the space of decision problems

Consider the following set of decision problems whose detailed description appears in Appendices A–D:

(1) preparing a telecommunication bid,
(2) organizing an advising plan,
(3) allocating research funds, and
(4) preventing school dropout.

These decision problems are very complex. Following their careful analysis, a set of shared problem characteristics emerges:

(1) These problems are highly dimensional: each case and solution are described by numerous characteristics.
(2) Solutions of these problems are based on a many-to-many mapping: a case is defined by many characteristics and its solution is also described by many characteristics.
(3) These problems are heterogeneous: different cases may lead to very different solution types.
(4) Solutions are context- and time-dependent: solutions to the same case may differ if solved for different contexts or in different times.
(5) Solutions are highly subjective: different professionals might end up with different solutions for the same case and context.
(6) Problem-solving is based on relatively few case characteristics: when professionals solve these problems, they often quickly identify few case characteristics as governing the particular case and proceed to solve the case with this view in mind.
(7) No codified knowledge exists: generally, there is no acceptable body of knowledge such as rules, laws, etc., which may assist professionals in solving these problems. Rather, knowledge lies in memories of experiences of past performance, whether successful or not.
(8) Future auditing or quality control may require explaining solutions: the solutions of these problems have long-term consequences that have to be implemented, revised, or abandoned. Consequently, one has to understand past decisions in order to make better future choices.
(9) Following characteristic (7) and the fact that there are no available detailed records of past solutions; practitioners are the sole source of knowledge. They have little time to spare in their work and little motivation to spend effort beyond their usual work to make their knowledge available to others; therefore, any solution devised must fit naturally into their present work practice.
(10) These problems demand solutions within a limited period of time. Stretching response time means damaging results (e.g., in school dropout) or total failure (e.g., in bidding).
(11) There is high cost to solution failure. These problems touch the lives of people and organizations; therefore, failure to address them may have serious consequences.

If this set captures the essence of these four problems, meaning that solving them requires addressing these characteristics; we might assume that their solutions also share many aspects. This assumption leads to stating the following hypotheses:

H1. It is possible to find a set of problem characteristics that is sufficient for describing diverse types of decision problems.
H2. It is possible to use these problem characteristics for sub-dividing decision problems into different classes.
H3. These classes would be meaningful for assisting the creation of DSS for these problems.

The first two are similar to Pearson and Shim’s hypotheses, but the third is an addition. We discuss and illustrate the three hypotheses through a
case study that, overall, follows a qualitative research method. The first two hypotheses are given support through an analysis of 22 decision problems. The third was checked with four examples, of which one is discussed. While this support is limited, we hope that it will be enticing enough to have other researchers explore the usefulness of these hypotheses through other case studies. Gradually, we hope to accumulate additional valuable supporting information.

3. Sufficiency of problem characteristics

The aforementioned problem characteristics are useful for describing several decision problems as discussed before. In order to check their usefulness for characterizing and organizing decision problems in general, we used WebGrid [12] as a knowledge acquisition tool to elicit knowledge on additional decision problems and their characteristics. WebGrid guiding mechanisms are especially suited for this purpose since they point their users to consider potential difficulties in the structure of the knowledge being elicited. For example, if two decision problems have similar characteristic values, WebGrid suggests that the user start thinking about a characteristic that would differentiate between the two problems. The results of a short session with WebGrid followed by cluster analysis are shown in Fig. 2. The decision problems (see Table 1) appear at the lower right part of the figure. The problem characteristics (see Table 2) appear on the sides of the value matrix, called a

![Fig. 2. An organization of decision problems and their characteristics with WebGrid.](image-url)
repertory grid. They provide the two extreme values of a characteristic, e.g., “high failure cost”–“low failure cost” values of the failure cost characteristic. The matrix shows the values of each problem characteristic for each decision problem on a scale of 1–10. The results of clustering are shown both for the problem characteristics and the decision problems as a tree connecting similar items. The scale above each clustering tree gives the level of similarity.

The process with WebGrid can continue and yield a more elaborate set of characteristics and problems, but for the purpose of illustration, the results shown in Fig. 2 are sufficient. Several interesting patterns already emerge:

(1) The correspondence between the list of characteristics from Section 2 and the grid characteristics is very good. The 11 initial characteristics translate into 11 slightly different characteristics and we added 5 more to account for issues that emerged in the knowledge acquisition process for differentiating between the different additional decision problems.

(2) In general, the results show that the characteristics are quite unrelated, suggesting that they are not interdependent. The principal compo-

Table 1
Decision problems used in the WebGrid model
(1) Plane engine diagnosis performed on the ground
(2) Preparing a telecommunication bid as performed in a particular Israeli firm (see Appendix A)
(3) Proposal DM: the decisions made when addressing a request for proposal
(4) School dropout DM: dropout problem in the Bureau of Education (see Appendix B)
(5) Organization advising plan: creating a plan addressing organization problems (see Appendix B)
(6) Trauma room DM: the decision-making in hospital trauma room
(7) Medical diagnosis: a decision-making by a doctor treating a patient
(8) Photocopier diagnosis: repairing a photocopier machine
(9) Design code standard conformance: verifying that a design adheres to design codes
(10) Fire fighting DM: decision-making during fire fighting
(11) System analysis DFD: analyzing systems using data flow diagrams
(12) Preparing house renovation bid: preparing a bid to renovate a house
(13) Software support help desk: answering customers problems about a software product
(14) Legal client prospects: estimating the chances of winning a legal case
(15) Consumer product help desk: answering customers’ queries about a consumer product
(16) Periodic medical checkup: decision-making during periodic medical checkup
(17) Lesson learning: accumulating organizational knowledge in the form of learned lessons
(18) Credit card evaluation: evaluating the suitability to provide credit to a customer
(19) University lecture scheduling: scheduling classes according to different restrictions
(20) Allocating research funds: selecting proposals for obtaining a particular level of funding (see Appendix C)
(21) Onboard plane engine diagnosis: diagnosis engine operation on board a plane
(22) Onsite environmental standard check: performing measurements to check the conformance to environmental standards

Table 2
Decision problem characteristics used in WebGrid model
(1) Group–Single decision: is it a group or a single user decision?
(2) Subjective–Objective: the level of subjectivity involved in the decisions
(3) Normal–Urgent DM: is an urgent solution needed or a regular one appropriate?
(4) Satisfactory data–Limited data: the amount of information/knowledge about the problem available for decision-making
(5) Context-dependent–Context-independent: are the problems or solution dependent on context?
(6) Many–Few cases: the number of past cases that could be consulted
(7) High–Low failure cost: the cost incurred when making the wrong decision
(8) Time dependency–Time independency: the influence of time on the problem or solution
(9) Procedure based–Experience based: the problem solving style
(10) Structure not important–Structure important: the internal relations between problem parameters and solutions is important
(11) Homogeneous–Heterogeneous: the uniformity of solutions to different cases
(12) Can–Cannot formalize: the ability to define the problem formally or structure it
(13) Normal–Minimal input: the data size required an input to the decision-making process
(14) Simple–Complex case description: the effort needed to describe a case
(15) Short–Long decision time: the time available for decisions
(16) Small–Large user group: the size of the DSS user community
ment analysis of WebGrid (see Fig. 3) shows few similarities (e.g., between “simple case description” and “structure not important”), but by and large, the characteristics are distributed quite evenly around the circle suggesting again that they are reasonably independent.

The two first items support the first hypothesis: It is possible through this process to find a set of meaningful characteristics for describing decision problems.

(3) There is an emergent distinction between “analysis” and “synthesis” problems even though there was no intention to expose these groups. In contrast, to prevent easy distinction, we did not use the “many-to-many” characteristic that is so profoundly typical of synthesis [29]. The distinction is critical because experience in engineering suggests that analysis problems are much more amenable to support, whereas synthesis is difficult to support. DSS research and implementation in synthesis is more likely to fail than in analysis where major successes have been developed [9,28]. In addition, inside the “analysis” cluster, there is an emergence of several groups: complex problems that require model-based, computationally intensive methods for their solutions; and groups of urgent, real-time, and “off line” diagnosis.

(4) Three problems described in Appendices A–D (except “allocating research resources”) are clustered together with three other general problems. This clustering makes much sense since they all seem to be difficult synthesis problems.

Fig. 3. Principal component analysis of WebGrid data.
The last two items suggest that the characteristics are good descriptors for classifying decision problems, hence supporting the second hypothesis.

(5) If we further use the results of the Cluster analysis in the Entail analysis [11] provided in WebGrid, we could further identify the problem characteristics that distinguish between the six different clusters (see Table 3) (in order to do so, we add a fictitious characteristic called “in-cluster”; assign the problems in each cluster a different value; and use it as an output characteristic for generating rules. Note that similar results could be obtained by the use of many other machine-learning programs). The rules are surprisingly simple and again, make sense. To illustrate, decision problems that involve heterogeneous cases are placed in the “synthesis” cluster. In addition, decision problems that could be formalized and have average failure cost are placed in the real-time diagnosis cluster.

(6) Other decision problems are placed in close proximity in ways that seem intuitively correct. For example, the group of the “urgent diagnosis” problems.

Several notes about the elicitation process are due. The evaluations of decision problems with respect to the characteristics were done by both authors separately and contrasted with each other. This was followed by a collaborative session of discussion and adjustments. There is some sensitivity of the results to the evaluations. This could be minimized by facilities that compare and contrast repertory grids of multiple experts [36]. We could also use AHP mechanisms of rating and consistency checking [34] for obtaining better consistent ratings.

To summarize this section, we consider the results of the WebGrid analysis as illustrating and supporting the first two hypotheses.

### 4. Solving one cluster problem

In order to demonstrate the third hypothesis, we show how we develop a DSS for one decision problem and how the rationale of the development is recorded in a diagram that can then serve to design DSS for similar problems. The DSS is developed for assisting in school dropout prevention (SDP) and the reuse of this solution is demonstrated for preparing a telecommunication bid. Fig. 4 shows the rationale behind the choices made in designing the DSS for the SDP. The figure captures how the initial limited understanding led to the problem characteristics, which in turn, led to intermediate conclusions that were translated into technologies and processes for developing the system. We now elaborate on the progress of the decisions as we move from left (top) to right (bottom) of the figure.

Given our initial lack of knowledge about this domain and the fact that such a future system would intervene in work processes of professionals, we decided to use an expert participatory development approach [31]. We did so in three different circles. The first circle included an expert attendance officer as part of the research team. The second circle included about 15 officers that helped us in focus groups, information collection, and method and system evaluation. The third circle included a highly experienced superintendent that provided overall guidance and feedback. This participatory development is

---

<table>
<thead>
<tr>
<th>Table 3: Rules characterizing clusters of decision problems</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Rules</strong></td>
</tr>
<tr>
<td>(1) If Heterogeneous then cluster 1</td>
</tr>
<tr>
<td>(2) If Structure unimportant and Few cases then cluster 2</td>
</tr>
<tr>
<td>(3) If Can formalize and Low failure cost then cluster 3</td>
</tr>
<tr>
<td>(4) If Objective and Few cases then cluster 4</td>
</tr>
<tr>
<td>(5) If Can formalize and Long decision time then cluster 4</td>
</tr>
<tr>
<td>(6) If Simple case description and Short decision time and Normal DM then cluster 5</td>
</tr>
<tr>
<td>(7) If Minimal input and Can formalize then cluster 5</td>
</tr>
<tr>
<td>(8) If Minimal input and Time independency then cluster 6</td>
</tr>
<tr>
<td>(9) If Minimal input and Cannot formalize and Simple case description then cluster 6</td>
</tr>
</tbody>
</table>
Fig. 4. Rationale of DSS for school dropout prevention.
central to our solution; it forced us to develop the system in a way that incorporates users’ input throughout the system life cycle. Following initial study of the decision processes and feedback from all the participants, we arrived at the characteristics of the problem. These are the 11 characteristics mentioned in Section 2.

The central decision—using case-based reasoning (CBR) [21]—is driven by the lack of codified knowledge and the inability to formalize such knowledge, and given practitioners’ reliance on past cases when they solve problems in their present practice. The latter, with the need to intervene the least in work practices, almost mandates the use of CBR. This immediately creates tasks for collecting case data since no record of such cases exist, and as a subtask, to create an agreed language between all project participants that will be used to document and represent cases. This was a difficult but necessary task. It became clear that no perfect agreement is possible but a workable compromise is achievable. Another subtask was to perform case quality control for making sure that all collected cases use the same language and include all necessary and correct information.

Another problem characteristic had major impact on the decision. The problem domain is heterogeneous: there are cases that are very different in the way they are addressed. Given this and the choice to use CBR, we decided to cluster the cases into classes that are more homogeneous. This is not new in CBR (e.g., Ref. [6]) and furthermore, is akin to attendance officers problem-solving: when they obtain a new case, they try to decide to which “class” of case it belongs.

From observing officers, we saw that they use little information in assigning the case to its typical class: in most cases, two to three problem properties were used. It was clear that this information is insufficient to drive the clustering we wanted to perform and to obtain meaningful support for decisions. Therefore, we decided that we needed to augment the information available on a case with additional knowledge. Since the source of knowledge was practitioners, we had to find a compromise: extract minimal useable knowledge that is part of the practitioners’ problem-solving and does not overburden them. The solution we developed is an influence matrix that records the influence of problem characteristics on actions taken to address the case [30]. The matrix has binary values and thus could be filled quite easily by practitioners. Since practitioners only fill the influences, the matrix is quite sparse. Consequently, it could be enlarged without burdening practitioners to include additional problem characteristics and actions. This facilitates future extensions and also addresses the context and time dependency of the problem domain. It is clear that different geographic places have different types of cases depending on demography and availability of measures for addressing the cases. It is also clear that the dynamics in Israel changes the nature of the problem over the years. The ability to extend the case representation by additional characteristics and actions provides partial response to these characteristics.

We mentioned that practitioners first classified new cases into typical classes using few case properties and subsequently, solved the case. In order to mimic this process, we clustered the cases using the influence matrix. This has yielded better results than the clustering based on case description. We developed a characterization of these clusters manually and by using a machine-learning program (WEKA; [40]). The automated approach was tested as a measure to reduce the subjectivity of the expert that characterized the clusters manually. Problem-solving thus became:

1. the user identifies two to three influencing case properties;
2. the user uses them to manually select typical classes (one or two) or the system does this automatically (using the learned classification rules);
3. the system uses influential case properties to retrieve most similar cases in the typical classes (usually two to three cases from a class); and
4. the user solves the case reusing and adopting past solutions and own knowledge.

The latter step was used to maintain the human influence in the decision cycle, making sure that system’s errors do not lead to mistakes that could impact the children involved. The system by definition was to be a support system. We performed successful testing in a lab setting with 12 officers [17].
As the project unfolded, it became apparent that the whole project is composed of two different stages. First, we could create a baseline system with reasonable quality that was tested successfully in a lab setting. This would enable practitioners to use and benefit from it immediately when we deploy the system. This initial development involved data collection, quality control, clustering, testing, test evaluation, and cluster correction. The expert attendance officer in our team had to spend much effort in this stage. This is prohibitive when deploying a full-scale system for an extended time period. Therefore, we had to develop mechanisms that would allow the system to grow and be maintained without the close monitoring of the expert and with the participation of its future users. The complete design maintains the overall structure of the system and introduces mechanisms from the area of recommender systems or collaborative filtering [7,33] to collect users’ feedback and evolve the case database [10]. We exercised this process by using the feedback we got from the lab tests with 12 officers; nevertheless, this stage remains to be tested in lab and real settings.

In summary, developing the school dropout prevention DSS contains the following two stages:

Stage 1: Developing a basic CBR system that is well examined, efficient, and capable of giving quality solutions to problems as characterized above. The system absorbs cases and knowledge from field experts. The knowledge enables organizing the cases into clusters that are subsequently characterized, and to locate in each sub-space the resembling cases that will contribute to the quality and efficiency of the solution. In developing the system, we can use the help of an expert during the process of structuring the problem (input and solution properties), characterizing and analyzing the clusters, and planning the experiments for testing the system.

Stage 2: Extending and maintaining the system by the practitioners’ community and a group of designated specialists for better quality management. In time, the accumulation of cases will improve the system’s quality and reduce the case’s built-in subjectivity.

5. Reusing solutions: the use of the architecture for a single remaining problem

Fig. 5 describes the rationale of building a DSS for “Preparing a telecommunication bid.” The changes (mark in numbers) from Fig. 4 are many and they cause several modifications to the mechanisms used although not to the overall architecture. These modifications depend on the specific project nature and organization but nevertheless, can describe the flexibility of our approach.

■ Few skilled people perform bidding while many field workers participate in dropout prevention (1). This leads to requiring accessible tools but not necessarily simple ones (4).

■ In bidding, a solution is required within a short time frame, in contrast to the possibility to delay decisions in dropout prevention (2). In order to address this, the solution process can be built to allow stopping at intermediate points with solutions at different levels of detail (5). In this way, bidding can always be given even if imperfect.

■ There are bidding scenarios accessible electronically even though not exactly in the required format (3). In contrast, there were few dropout cases that were available for system building.

■ When using CBR, a need arises to add new cases that include documentation in the required format (6). This aspect was omitted from Fig. 4 but had no effect on the outcome. Here, it is introduced explicitly. The need to document, and the need not to interfere with users work, mandates the use of a simple representation that can easily be accommodated into practice. Therefore, the same simple representation of knowledge arises in bidding but due to a different reason than in dropout prevention.

■ The need to cluster cases and the new way of multi-resolution decomposition (5), leads to using hierarchical clustering (7) [16,29]. The matrix relation that will be generated for each case (9) drives this clustering.

■ Given the availability of cases, they need to be augmented with the required information and new cases collected (8).

■ The solution process is different from the dropout prevention. In bidding, the cluster is found at the
Fig. 5. Rationale of DSS for preparing a telecommunication bid.
most appropriate level of detail and is used to assist in the bidding process (item1).

In addition to the change mentioned, there are characteristics that appear in the two problems with slightly different importance levels. For example, the characteristic “minimal input” exists in both problems, but in dropout prevention, the lack of input is more significant. This difference can be noticed in the WebGrid model (Fig. 2) where characteristics are weighted on a 1–10 scale. In our example “minimal input” got 7 in dropout prevention problem and 4 in preparing a telecommunication bid problem.

There could be variations on the solutions. For example, since a bidding system will work for one company with a limited users group, the addition of new cases is expected to be at a slow rate. Therefore, quality control could be done by appointing one person to be in charge of quality control instead of using collaborative filtering. These multiple paths could be explored and also represented by such influence graphs.

6. A DSS for planning DSS program

One interesting implication of this work emerges when we consider developing a DSS that will assist practitioners in planning different DSS projects; we denote such a system by DSS2. In order to gain some insight, we incorporated two additional decision

![Fig. 6. An organization of an extended set of decision problems with WebGrid.](image-url)
problems into the previously analyzed collection: “Planning a DSS program” and “Developing consumer product concepts.” These are both highly complex, open-ended, synthesis problems. Their ranking over the decision problem characteristics and clustering with WebGrid, places them with lesson learning—another complex problem (see Fig. 6).

The other synthesis problems are placed next to the new cluster. According to our hypotheses, we could reuse some solution aspects from related solutions for addressing the new problems. One significant aspect in solving the school dropout problem is the incorporation of knowledge in the form of influence matrices that are also used to cluster the solutions for using effective CBR as the solution backbone. Such an influence matrix could be used in DSS to model the relationships between case input properties (i.e., problem characteristics) and case output properties (i.e., DSS architecture description). Since we use an influence graph to detail solution processes of building DSS, we have the data to create such matrices [30]. For example, in the case of the school dropout prevention system, the matrix would be as shown in Fig. 7, and Fig. 8.
shows the influence matrix of preparing a telecommunication bid. The changes in the properties are underlined and the changes in the relations are colored in dark gray. We can easily notice that the matrices are quite similar; their similarity is 0.78 (according to Ref. [5]: “max common”=22 light gray common cells divided to “max matrix”=28, and assuming that “Diverse case collection” is equal to “Case documentation” since they have similar effects on the process). Note that the columns and rows need not be the same in both matrices or in the same order to allow for calculating the similarity.

Once we have created many systems and recorded their processes with influence graphs and subsequently matrices, we could use these matrices to cluster the DSS cases. We expect the results of this clustering to be better than the clustering based solely on case properties done presently with WebGrid. We also expect that this clustering will be able to provide guidance into building DSS at least as well as the clustering created with WebGrid.

Since the “planning of DSS program” is in a neighbor cluster of the cluster with the SDP and the “preparing a telecommunication bid” problems, we might expect to have similarities but also deviations between the solutions to these three problems. One such difference might be that due to the increased complexity of the planning DSS program compared to SDP, that it would not be sufficient to cluster cases based on the matrices as we showed above. In contrast, a better clustering and subsequent CBR implementation might have to rely on manipulating the original influence graphs without transforming them to matrices. In order to implement this, we would simply need a similarity measure between such graphs, which is available using graph edit operations [24]. Another difference between bidding and planning of a DSS program is that the former is time-limited while the latter is less constrained in this regard. Therefore, we might be able to use one-level clusters as in dropout prevention instead of the hierarchical, multi-resolution clustering in the bidding solution.

We see that our model opens up a new perspective in looking at decision problems and combining knowledge created in various DSS projects into a knowledge structure that could assist systems developers in future tasks.

7. Discussion and conclusions

Designing a DSS is a complex task prone to errors and susceptible to colossal failures. Contemporary analysis of the space of decision problems and their DSS solutions is preliminary. This could lead to making sub-optimal development choices and ignoring previous experience, thus increasing the chances of failure. Having a framework that provides accessible and easy to use guidance for making judicious development choices can have a positive impact on project development prospects. Consequently, we put forward a framework and three hypotheses that improve the description, reporting, and reuse of information about the implementation of DSS for different decision problems. We provided initial support for these hypotheses but their practical usefulness remains to be tested empirically through a collective effort.

In relation to the first two hypotheses (H1 and H2): we created a set of characteristics that could be used as a core and evolving set for describing decision problems. This characterization could be used to cluster the space of decision problems in a way that corresponds to our intuition. Furthermore, upon inspecting the clusters that emerged from preliminary data, we see that cases within clusters tend to have similar or related solutions. The classification to clusters would be dynamic as existing problem descriptions are refined and additional decision problems are added to the analysis. In addition, the classification would be context-dependent, i.e., it would depend on the particular experience of its creators and their use of language in creating the classification.

In relation to the third hypothesis (H3): we illustrated one such cluster. We showed how a system’s architecture and development process for SDP arise from these characteristics. We developed this system and tested it successfully in a lab setting. We think that when developing DSS for decision problems that share similar problem characteristics, parts of the solution architecture and its details could be reused. We illustrated this by the preliminary design of a DSS for telecommunication bidding.

Our experience was that practitioners that took part in developing the SDP DSS found their role to be reasonably easy and reported improved performance just by being aware of the approach and without even using the classification. We contend that similar to
these practitioners, developers of DSS could benefit from the framework even if they only use parts of it. For example, exercising models such as those in Figs. 4 or 5 could improve the traceability of decisions and prevent overlooking important problem characteristics. We further contend that the SDP DSS experience suggests that the proposed approach is useable and rewarding, therefore could be practically feasible.

Another illustration of the third hypothesis arises from considering DSS^2, a DSS for the planning of a DSS program. In order to get an initial idea about how DSS^2 could look, we introduced this task description into WebGrid and found a new clustering of all the decision problems. We showed how information about the decision problems in “close” proximity to the planning a DSS problem could be reused to propose building blocks for designing DSS^2. One such building block is using CBR with clustering of previous cases of DSS developments. We showed influence matrices of two DSS that serve as the input to such clustering and similarity calculations.

The framework we propose could best serve researchers and practitioners if the reports of DSS projects, whether in practice or as case studies in the research literature, would follow the structure presented in this paper. In this way, additional data would be accumulated and become available for refining the classification of the space of decision problems. This in turn would improve the support for building new DSS. Presently, most such case studies (e.g., Refs. [20,35] are not being used in practice due to the excess effort required to sift through their data and extract directly relevant recommendations.

In summary, we propose that the space of decision problems could be organized in order to shed light on its structure, which in turn could be exploited when developing future DSS. This proposition remains to be fully tested by having different developers and authors adopt the influence graph convention (e.g., Figs. 4 and 7 or Figs. 5 and 8) for reporting the development of DSS, thus allowing the accumulation of information about such projects.

Appendix A. Preparing a telecommunication bid

A telecommunication firm needs to prepare bids for support software on a regular basis. The software is a complex structured product that has operational demands depending mainly on its target environment (physical, users, etc.). The initial knowledge for preparing the bid could be minimal, including general specifications of the product, main demands, budget level, schedule limitations, functional demands, physical limitations, etc. The subjectivity level at this stage is high. The bid creator views the specifications from a personal perspective with limited ability and time to obtain additional clarifications.

The firm has a team that specializes in bids, however, it needs the knowledge provided by field engineers and developers. If the bid is too complex, experts could be consulted to resolve particular issues. The team will attempt to locate bids of similar past projects and reuse them in the current project.

Usually the schedule is tight and a suitable bid has to be created using accessible tools. A bad bid or a bid that does not fit the need may cause great losses to the firm or lead to losing the project to competitors.

In an interview with a representative of a telecommunication company handling many bids the problem was characterized as follows:

The bid must answer many functional demands. In many cases, there is little information on these demands or on the customer. For example, in setting up a telecommunication system for a communication company, we lack necessary information about the number of subscribers, the number of calls per day, high activity hours during the day, form of communication (cable, cellular, SMS), amount of data storage, etc. The lack of this data that could influence the system’s configuration, makes it very difficult to come up with the proper bid. In addition, lack of information on the growth rate of the company makes it difficult to create a system that can expand and support such expansion.

The time factor is very important since the bid has to be submitted before the deadline. Bids are often subjective: in some cases, two companies with similar profiles would receive different offers from different planners, in other cases, one company could receive two different offers (in different times) from two different planners. The environment and geographic place influence the nature of bids as well. In the communication field,
there is a big difference among subscribers in different parts of the world. In the Far East, many use SMS technology and less vocal calls, a fact that dramatically affects the offered system configuration and bid. The resources put into a bid depend on its magnitude and the chances of winning it with minimal future risks. There is a conflict of interests between the will to invest in the bid in order to make it profitable and increase its chances and the company’s willingness to invest in bids that do not win. Therefore, a very efficient, effective, and qualitative process is demanded for offering bids. In any case, the price of failing in a bid is significant.

This characterization suits the type of problem described and intensifies the need for qualitative and applicable solution.

Appendix B. Organization advising plan

An organizational advisor is invited to offer a proposal for an intervention plan for addressing an organizational problem (e.g., bad financial situation, management issues, troubled human relations). The advisor examines the organization and its context and analyzes the various interacting aspects.

The initial assessment based on meetings with different managers is very subjective; each describes the problem from his/her point of view in an attempt to prove that his/her actions were correct and the problem originated elsewhere. The knowledge received is partial and it focuses on the general description of the problems, macro-description of the organization’s structure, goals, and aims. Hidden problems such as power conflicts, executive problems, and bad investments might not surface at this stage. The organizational advisor is the specialist that performs the job. He can use the help of additional workers from his firm having lower or higher level of expertise. After the initial analysis, the advisor recalls successful organizational interventions carried out in similar organizations and tries to adopt similar solutions, which may prove to be suitting, efficient, and successful.

Usually the organization is under time pressure to solve its problems and the advisor is asked to offer a plan in a short period of time and the plan is expected to be effective and quick to implement. An unsuccessful plan could be critical to the organization.

Appendix C. Allocating research funds

Funding agencies receive different proposals in diverse and alternating topics and must decide which proposal to support and which to decline. The proposals have a different level of complexity, use different methods, and address varying issues and areas. There is always a risk that a proposal would miss its stated goals.

There is no “objective” method to analyze a proposal and give an exact and unbiased consideration. The decision-makers, a team of experts, do not always have the proper knowledge needed to examine a proposal given that each expert specializes in specific subjects. Thus, the degree of subjectivity is quite high. External qualified specialists or reviewers who give their opinions are also consulted, but nevertheless, the committee makes the final decision.

Each proposal contains basic information and hypothesis that may not be verified or lead to the goals of the proposed research. It is likely that the members of the committee (the experts) will try to retrieve similar proposals/researches in the past and infer from these proposals to the current offer. Usually the committee is under time pressure. The number of proposals is large and the decision time is limited. The tools used by the committee are simple and accessible (grading, weighing) and the decisions should be effective and well understood. Allocated funds often mean the survival or ending of projects. Successive wrong decisions may result in waste of resources and damage to the fund’s reputation.

Appendix D. Dropout problem in the Bureau of Education

Close to 9% of the students in Israel drop out of school. In the year 2001 as much as 25% of 12th grade students dropped out. This is a significant problem in education and its context and the solution is in the form of a decision support system for attendance officers.
School dropout can be seen as a systemic problem with both human and social aspects. Attendance officers, who enforce the education attendance laws, find themselves dealing with many different cases in a situation with limited resources. Also, creative solutions conceived by one attendance officer are seldom shared by others.

Environment, city hall policy, education policy, and demographic features influence the attendance officer’s work. Therefore, solutions for one area may fail somewhere else. There is no coherent work methodology and the attendance officer has to apply good judgment, use experience or colleagues (usually more senior and experienced) assistance, in order to arrive at efficient and effective solutions. In addition, the level of subjectivity of the attendance officer’s decision is very high and it seems that each officer focuses on another aspect (constitutional, procedural, tendency to psychological treatment, emphasis on relocating the student, etc.). Usually the initial information on the problem is scarce and the officer focuses on two or three distinguishing aspects of the case (violence, special education, psychological diagnosis, etc.). These aspects help him/her devise a solution.

Attendance officers’ tools should be relatively simple and friendly, so that during a short period of time they would be able to handle well and efficiently the variety of problems they face. As a result of the importance of the problem and the human factor, it is important to arrive at efficient solutions in a reasonable time that would stop a senseless dropout. Bad solutions may cause irreversible damage!

References


