Supporting Modeling and Problem Solving from Precedent Experiences: The Role of Workflows and Case-Based Reasoning

Daniel S. Kaster   Claudia B. Medeiros   Heloisa V. Rocha*

Abstract

Environmental planners take advantage of Spatial Decision Support Systems (SDSS) to deal with data and models for problem solving. However, these kinds of software usually provide generic models, which require considerable effort to be specialized to fit particular situations. This paper explores a solution which couples Case-Based Reasoning (CBR) to an existing SDSS, named WOODSS, to help planners to profit from others' experiences. WOODSS is based on a Geographic Information System, and interactively documents planners' modeling activities by means of scientific workflows, that are stored in a database. This paper described how CBR has been used as part of WOODSS' retrieval and storage mechanisms, to identify similar models to reuse in new decision processes. This adds a new dimension to the functionality of available SDSS.

Keywords: Environmental decision support, Case-based reasoning, Scientific workflows, GIS

1 Introduction

Decision Support Systems (DSS) are software that help users apply analytical and scientific methods to decision making [6]. DSS that focus on the environmental domain are referred to as Environmental or Spatial DSS (EDSS/SDSS), providing analysis tools to handle spatio-temporal data found in environmental processes [41, 16]. An environmental simulation model may be defined as a computer-based technique to imitate, or simulate, the behavior and the reactions of various kinds of real-world processes [45].

EDSS/SDSS must provide support for model specification and construction. However, they usually provide only generic models, which need to be adapted to fit particular situations. This requires considerable effort and expertise, which includes the appropriate choice of models, and of data to instantiate them. Indeed, model suitability and data selection are sensitive to the geographic

**Address:* IC-UNICAMP, CP 6176, 13081-970 Campinas-SP, Brazil. Email: dkaster@dc.uel.br,{cmbm—heloisa}@ic.unicamp.br

context, and often depend on the region and on the environmental constraints for which the solution scenarios are being built.

The construction of solutions usually requires cross-disciplinary work and is reached only after intensive collaboration of groups of experts. However, decision making processes are frequently performed in an *ad hoc* manner, with insufficient documentation and very little support for interchange of expertise among groups of planners. Thus, a considerable amount of time is spent in reinventing solutions to problems, while money and time would be saved in profiting from past experience.

This paper discusses a software tool under development at the Institute of Computing of the University of Campinas (UNICAMP), Brazil. The goal of this tool is to help decision makers in the environmental domain to collaboratively exchange their experience, and profit from learning about past solutions to similar problems. This tool, named WOODSS (WOrkflOw-based spatial Decision Support System), works in conjunction with a Geographic Information System (GIS) and is based on two concepts: (i) the use of scientific workflows [47, 2] to represent environmental models that decision makers have designed and (ii) several kinds of retrieval mechanisms to help users choose the most adequate models from those available in the WOODSS database. In special, the work presented in this paper concerns the the use of Case-Based Reasoning (CBR) [38] as a retrieval mechanism.

This approach combines work on database systems, artificial intelligence and workflows. The database contribution lies in managing WOODSS' modelbase using database techniques. Classical architectures for decision support systems consider two kinds of storage entities, managed separately: the Modelbase, where models are stored; and the Database, containing field data, metadata and administrative information. In WOODSS, both Database and Modelbase storage units are handled in a unified way within a single database management system. This allows adopting compact storage policies, as well as flexibility in model handling, with support to update and expansion of the modelbase. Models are represented as scientific workflows, stored in this base, and can be progressively enhanced and combined.

Artificial intelligence research is used in the context of CBR, whose retrieval techniques are added to the retrieval mechanisms of WOODSS, offering context sensitive similarity analysis. Decision makers can either reuse existing models, or combine/adapt them to their specific needs, thereby solving problems incrementally.

The main contributions of this research are: (a) a discussion of the theoretical and pratical capabilities of CBR in environmental decision support; (b) analysis of the process of eliciting requirements for using CBR in this domain; and (c) presentation of implementation issues concerning the combination of CBR and scientific workflows in WOODSS.

The rest of this paper is organized as follows. Section 2 presents an overview of related work by discussing the applicability of CBR in environmental modeling and decision support. Section 3 introduces WOODSS. Section 4 presents the use of CBR in WOODSS, showing the CBR schemes adopted in this work.

Section 5 illustrates implementation issues through a pratical example. Finally, section 6 presents conclusions and future work.

2 CBR in environmental modeling and decision support

2.1 An overview of case-based reasoning

Case-Based Reasoning (CBR) is a model of reasoning which consists in solving new problems by adapting solutions that were used to solve old problems [38]. CBR research is tightly connected with artificial intelligence, within the domain of knowledge management [48].

The principle of CBR is based on a cognitive model named Dynamic Memory [43]. This model states that human memory is dynamic because it is continuously changing according to the new experiences one is exposed to. These individual experiences, or *cases* in the CBR terminology, encompass lessons learned in a specific context, which can be used to face new situations. Thus, knowledge in CBR is embedded into particular cases, and in their interrelationships.

A *case* is a contextualized piece of knowledge representing an experience [30]. It can be for instance an account of an event, a story, or some record. Even though there is a lack of consensus in the CBR community as to what to represent in a case, its description typically comprises at least:

- The *problem*, that states a case and describes the state of the world when it occurred; and
- The *solution*, that states the solution derived for that problem.

The basic processing cycle of CBR comprises four tasks (nicknamed the *four REs*) [1], as illustrated in figure 1. This cycle assumes that there exists a case "memory" (the Case Base) that contains knowledge of situations/cases previously encountered. The cycle consists of iteratively executing the following steps, given a problem to be solved:

1. REtrieve from the Case Base the set of cases most similar to the input problem;
2. REuse the solutions of these retrieved cases. If necessary, adapt their solution to solve the input problem, thereby creating a solution tailored to it;
3. REvise the correctness and usefulness of the solution adopted in step 2; and
4. REtain the new solution in the Case Base as part of this new case, for future utilization.

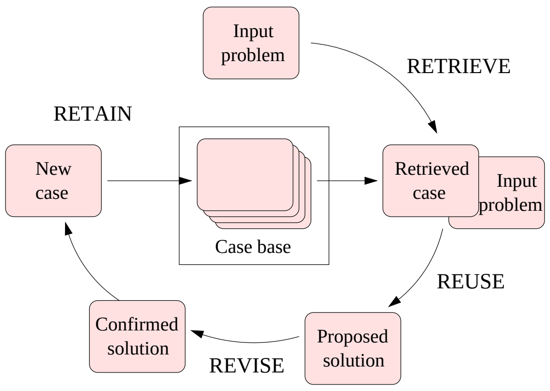


Figure 1: Basic processing cycle of CBR: The "four RE".

There is a whole range of CBR systems proposed in the literature, At the two extremes are *fully automated* systems and *retrieval-only* systems [30]. Fully automated systems are those that solve problems without user intervention and have some means of interacting with the world (e.g. sensors) to receive feedback on their decisions. Retrieval-only systems work interactively with a person to solve a problem. Their role is to augment a person's knowledge, providing cases for the person to consider that he or she might not be aware of; the person will be responsible for the actual decisions. There are very few fully automated CBR systems, all of which focused on very strict domains. Most CBR tools usually require human intervention, especially in the reuse phase [49].

2.2 CBR applicability in modeling and decision support

This section discusses the theoretical suitability of using CBR in environmental modeling and decision support. A close parallel can be traced between the way experts build models and the CBR process. First of all, model development usually fits the basic principle of CBR, which consists in *recurring* to old situations to solve a problem. Environmental modeling relies heavily on past experience, frequently improving old models and/or assembling parts of them to construct new ones.

Second, because of the inherent complexity of natural processes, experts usually build environmental models founded on experimental results and hands-on experience. Even though some generalizations may be suitable, many situations represent exceptions. By the same token, in CBR knowledge is not formally modeled, but it is defined extensively, through a *set of instances* [1, 35]. Each case is stored in a Case Base and individually represents knowledge on a specific instance of the modeled problem. A set of cases may be needed to encompass the domain knowledge needed to solve the given problem. Thus, CBR offers the flexibility required by environmental modeling, storing instances of random processes, and states in distinct instants.

Besides its suitability to modeling, CBR is also interesting in decision support because it intrinsically deals with problem solving. Knowledge acquisition, sometimes called knowlegedge harvesting [31], is part of the process of problem solving. Improving knowledge of an environmental process in the CBR context can be acommodated by storing new cases in the Case Base, to be retrieved subsequently to solve similar problems. Furthermore, CBR can be employed not only in problem solving, but also in experimentation - e.g. in simulation of scenarios.

CBR applications can be broadly classified into two main types: (i) classification tasks, and (ii) synthesis tasks.

Classification tasks cover a wide range of applications which are primarily concerned with case retrieval. This encompasses prediction, assessment, diagnosis and process control. Examples of classification CBR applications in the environmental context are the CAse-based Range Management Adviser (CARMA) [7], which deals with grasshopper infestations, the Water Quality Simulation Module (WQSM) [46], that controls oxygen emissions in a plant waste water treatment environment, the ZONATION system [23], which is focused on soil classification, and the NEMO system [25], for air quality prediction in urban areas. Fuzzy techniques can also be employed in this context – e.g., [39], for weather prediction.

Synthesis tasks attempt to create a new solution by combining parts of previous solutions, usually in design, planning and configuration applications. Most systems of this type must make use of adaptation and are commonly hybrid systems, combining CBR with other techniques. Examples of these kind of tools for environmental problems are CHARADE [37] and CARICA [3], which aim at supporting the user in the process of fire-fighting including both situation assessment, planning activities, and training support for fire-fighting intervention. Related work in the environmental context concerns developing CBR-associated techniques for knowledge capture and reuse – e.g., the work of [50, 8] concerning satellite image retrieval for spatial knowledge management.

Finally, CBR can be used to provide mechanisms to help decision makers choose the right models to apply according to the problem faced. This is the main motivation that is behind the use of CBR in our research. The subsequent sections present WOODSS and its use of CBR.

3 The WOODSS system

The WOODSS system (WOrkflOw-based spatial Decision Support System) [44] is an open and extensible SDSS which is being built at the LIS laboratory in UNICAMP, Brazil. It is founded on two basic concepts: the use of scientific workflows to document decision support processes in the environmental domain, and the premise that in this domain decision making processes are sequences of activities that lead to the production of a set of maps as a fundamental part of the planning output. Each map in the set may reflect an alternative solution scenario for a given problem, or the map set may illustrate complementary actions to be taken in a given situation. The map set is accompanied by a set of documents that contain directives and policies to be enacted in the region depicted therein. The enactment of the plan consists in implementing these policies in the region, following the spatial constraints shown in the maps.

A spatial decision process is prompted by a problem where location and spatial distribution are relevant factors. This process is solved by combining distinct kinds of spatial analyses on the region of interest, associated with other non-spatial information. The process can be described as a sequence of steps involving creation and combination of maps, to produce a final set of maps. This whole cycle may be repeated over the years for a given region – e.g., as the plan is enacted, environmental changes will prompt plan revisions, and thus new maps and directions.

For spatial analysis and map production, Spatial DSS (SDSS) rely on a Geographic Information System (GIS). For the purpose of this paper, it suffices to assume that a GIS is an information system that manages geographic data and provides sophisticated spatial analysis and visualization tools, including cartographic renderings. Geographic data may comprise maps, satellite and radar images, and a variety of datasets and attributes about phenomena of given regions.

3.1 Using scientific workflows to document models and cases

From a SDSS perspective, each map resulting from a spatial decision process is the result of the execution of a spatial model. Under this perspective, models can be specified in terms of their inputs (constraints, data files and parameter values), outputs (materialized in maps) and execution states [40]. Map generation using a GIS is a partially ordered sequence of activities, followed by sucessive adjustments, using the GIS' functions [44].

This structure for organizing the solution of a decision problem is very similar to that of workflows. The notion of a workflow originated in business domains to specify the "flow of work" to execute business procedures. A workflow can be defined as a set of tasks involved in a procedure along with their interdependencies, inputs and outputs. Each task is called an *activity*, and can be executed by one or more *agents*, in a given *role*; thus, an activity is a unit of work[22].

Environmental decision making has a strong component of empirical experimentation, with a long cycle of sucessive approximations through trial-and-error. Traditional workflows do not provide such flexibility in specification. The idea is to use a special kind of workflow to support the representation of spatial decision making processes – the *scientific workflow.*

Scientific workflows [47, 2] are a special kind of workflow suited to documenting and specifying scientific experimental activities within a laboratory. Whereas office workflows are concerned about goals, scientific workflows are centered on data and process management. Scientific work is characterized by a great degree of flexibility and presents a much higher amount of uncertainty and exceptions than business work. Scientific workflows are concerned with supporting the activities of collecting, generating, and analyzing large amounts of heterogeneous data, obtained from various experiments, models and statistical analyses; furthermore, they describe the experiments themselves. In particular, their execution must allow deviation from the specification while the workflow is being executed, as is often the case in scientific experiments.

While having the same basic components of business workflows, the specification and management of scientific workflows allows undoing activities, unfinished executions and on-the-fly ad-hoc specification (e.g., letting a workflow start execution before its specification is finished). In order to achieve these goals, the specification of a scientific workflow must describe the data sources used, the processes activated, their temporal and data relationships, their execution constraints, the agents involved and their roles. Furthermore, they can be updated during their execution, which requires implementation of specific execution control and monitoring techniques [2].

The work of [44] shows how environmental decision processes can be adequately modeled in terms of scientific workflows. These workflows offer a clean view of the whole decision process, helping experts to control and plan the next steps of the process. Furthermore, they can facilitate model execution, because workflows are parameterized specifications, and can control the interactions of distinct analysis tools. With the emergence of the notion of Semantic Web and Web Information Systems [17], scientific workflows are being disseminated as the means to organize cooperative scientific experiments – e.g. in geosciences [9] or in bioinformatics [5]. Furthermore, they are being widely used in the context of Web scientific computation and Grid architectures - e.g., [32].

3.2 Overview of WOODSS

WOODSS aims to provide mechanisms to handle and manage models, supporting the development of decision solutions. WOODSS interacts with a Geographical Information System (GIS) by capturing user interactions with the GIS and documenting them by means of scientific workflows. These workflows are used by WOODSS in three roles: (i) as a means for documenting a decision process; (ii) as high-level specifications of an environmental simulation model; and (iii) as executable parametrized specifications of decision procedures, which can be reused and adapted for similar situations.

WOODSS' modelbase is a database that stores scientific workflows and associated data. Decision makers can query this modelbase to retrieve the models most relevant to the problem faced, and reuse or adapt them, by adding or removing activities, changing the flow of data among these activities, or resetting their internal parameter values. Furthermore, workflows can be directly launched to execute in the coupled GIS. In this sense, besides documenting decision processes on the fly, the storing of new workflows in WOODSS also constitutes a means of progressively enriching the modelbase.

Figure 2 illustrates WOODSS' three kinds of user interaction modes: (1) user-GIS; (2) user-WOODSS; and (1-2) user-GIS-WOODSS, which combines

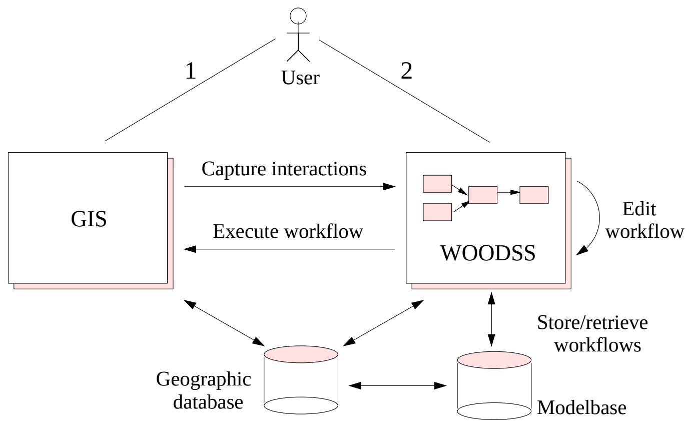


Figure 2: User interactions with WOODSS.

the previous ones. In the first mode, users ignore WOODSS and interact with the GIS, to define and implement some model, whose execution is materialized into a map. WOODSS monitors this interaction in real time and generates the corresponding scientific workflow specification. In the second interaction mode, also called workflow construction/editing, users access WOODSS to query its modelbase and update and/or combine retrieved models, constructing a new model, which can be subsequently executed in the GIS. Typically, this second kind of interaction mode occurs when decision makers want to find out about previous solutions to a similar problem. These kinds of interaction enact the three main goals of WOODSS: (a) documentation of decision processes; (b) support for decision making; and (c) modelbase construction.

The architecture of WOODSS, conceptually illustrated in figure 3, depicts its main functional modules: Interface, Monitor, Update, Query and Workflow Manager. The Monitor captures users' interactions with the GIS, translating them into a workflow which is passed on to the Workflow Manager. The latter is responsible for managing the modelbase. The Interface allows decision makers to graphically visualize and create planning processes and models in terms of workflows. It mediates user requests for browsing (Query module) and update (Update module) the modelbase.

The figure shows two distinct storage units - Modelbase and Geographic database. They are shown separately to distinguish the kinds of data stored. However, unlike standard decision support system architectures, both modelbase and database are handled together by a single relational database management system. This improves system performance and helps users in managing their models, which are associated with the relevant geographic data.

More details about the functionality and design of WOODSS are outside the scope of this paper, and can be found in [44]. The next section concentrates on describing how WOODSS has been enhanced with CBR.

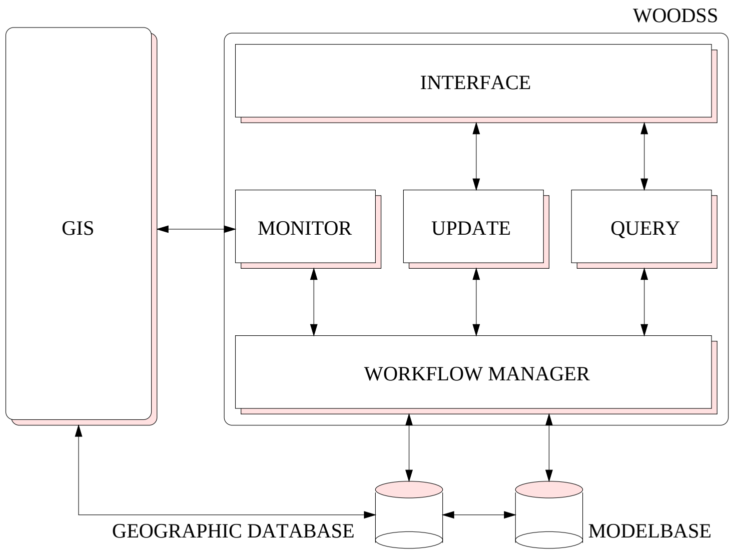


Figure 3: Architecture of WOODSS.

4 CBR in WOODSS

The first version of WOODSS, described in [44], did not count on CBR. It offered model development and management facilities to improve knowledge reuse. However, retrieval facilities were limited, and users had no help in identifying the model to use in a given situation. The model retrieval mechanism was restricted to keyword match, which could only identify exact (syntax) matches. Thus, in order to find some solution to a problem, a user would have to know what keywords had been employed in defining this problem and associated workflows.

Keyword match has several other limitations. One is that it ignores simple correspondences, e.g. synonyms and concept abstractions. This limitation is enhanced when groups of experts want to share experiences (e.g. see comments in [9]). They may use slightly different vocabularies and distinct points of view to document their models. Another limitation is that all features of a given problem have the same weight, while most times it is necessary to prioritize features when describing a problem [27, 28].

The approach investigated to overcome these limitations was to integrate CBR mechanisms into WOODSS to help users choose the most useful models (or parts thereof) at the right time. Thus, this work concentrated in the *retrieval* and *retainment* phases of the CBR process (which are complementary to each other). The main issues to be faced in these steps are case representation, indexing, similarity analysis and storage and retrieval algorithms, each of which is discussed in the subsequent sections.

4.1 Case representation

The first implementation issue, from a CBR point of view, is how to represent a case in a computer system to promote better reuse. Related work mentions "case memory" and "case database" to store cases, without detailing the representation structures adopted. We recall that a case is a contextualized piece of knowledge representing an experienced situation, and is usually formed by two components: *problem description* and *solution*.

Our approach, as will be seen, consisted in representing each case by a pair $<$ scientific workflow, associated metadata $>$. The workflow specifies the model – the *solution component* used to solve a problem – while the metadata describe the problem and the parameters used to instantiate the workflow. From an internal implementation point of view, this pair corresponds to a set of pointers (i.e., database record identifiers) – one pointer to a workflow, and three other pointers to associated metadata records. Workflow and metadata records are, in turn, stored in database relations. There are five main relations to manage workflow data (Activity, Data, Dependency, Workflow, Actor). Metadata records are explained at the end of this section, and detailed in 4.2. Further details on the database schema appear in [26].

Since a WOODSS workflow can be seen as a graph that represents an algorithmic execution, the first idea was to use the workflow's topology to help derive its semantics. This was soon abandoned, because though workflows have a well-defined structure, this structure does not clearly denote the semantics of the implicit decision process, because the same workflow patterns occur for distinct purposes.

Figure 4 gives an example of why workflow topology does not suffice to identify a case. Consider the generic problem of deciding on planting strategies in agricultural planning, where management practices must preserve environmental conditions. This kind of activity is called agri-environmental planning. Suppose that part of the plan involves solving the subproblem portrayed in Figure 4(a): "determine the most appropriate areas in a region to plant a crop x, given that its best yield is for soil type y, and where the potassium rate in the soil moisture is within a range of $z\%$". This problem has two constraints - soil type and potassium rate. Its solution requires identifying the areas which satisfy each constraint. In a GIS-based DSS, this is achieved by an operation called classification, which partitions a region according to some predicate. In this case, Figure 4(a) shows classifications based on soil type and on potassium content, resulting in two different maps – Soil', K' – for the same region. The combination of both constraints is achieved, in a GIS, by using the *overlay* operation, which provides as output a map where the areas satisfying both constraints are outlined. Figure 4(b) shows a solution to another subproblem whose statement is: "determine the appropriate recommendation of potassium fertilizer over a given region in order to plant crop x, considering that the region presents spatial variability both of the potassium rate and soil types". Here, it is necessary to determine the demanding fertilizer quantity (map reclassification based on the nutrient rate), and adjust it to each soil type (weighted map

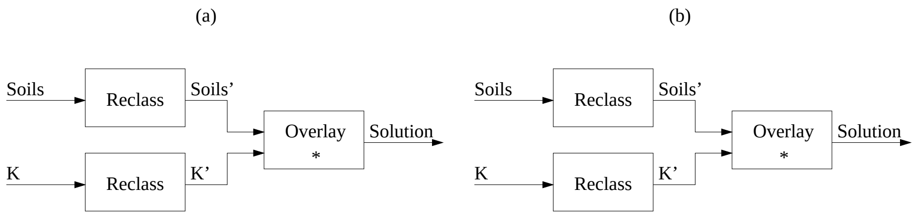


Figure 4: Topologically identical workflows but with different purposes.

overlay).

Both workflows in Figure 4(a) and 4(b) have the same topology and use the same inputs (soil map and potassium spatial distribution map). Both use the same GIS operations (*reclass* and *overlay*). However, the goals are distinct. In other words, different – though related – problems can be solved/implemented very similarly, and thus their workflows will be essentially identical, but answering to very different needs. By the same token, a given environmental-related problem may have very different solutions, depending on the region under study and the experts involved. In this case, the resulting workflows/models will be completely different, and their topology cannot be used in case-based retrieval.

As a consequence, we opted to adopt metadata to define the *problem component* of a case. These metadata are associated to each workflow to state the problem focused and describe its meaning. The drawback to this solution is that users must intervene to provide the appropriate metadata, to enhance workflow semantics. On the other hand, it is semantically richer than alternatives that consider automatic metadata extraction.

There remained the issue of defining the appropriate metadata components to describe a problem. The set of attributes chosen combine two kinds of knowledge: features elicited from common descriptions available in the environmental decision making field; and a classification of the most common queries during environmental decision making. This classification divides experts' concerns into the following categories [44]:

- Area-based queries. The basic predicate specifies a region. Users search for decision processes and/or models executed involving a specific geographic area. This allows finding out, for instance, all studies conducted in a given region.

- Problem-based queries. The predicate concerns a problem. Users look for decision processes developed to solve problems with similar objectives, to take advantage of experience acquired in the resolution of the same problem over other geographic areas. This allows retrieving, for instance, all solutions stored concerning nutrient balance for a given type of soil and crop.

- Process-based queries. The predicate involves a model, and users request to see its distinct implementations. This could show possible variations in the solutions of a problem, with their causes and consequences – e.g. the kinds of products used to implement a given nutrient balance procedure.

We point out that there are already several metadata standards proposed for the environmental domain (e.g. FGDC [15] or [34], which is suited for ecological sciences). The main difference is that these standards adopt atomic (non descriptive) metadata fields. We instead opted for two kinds of metadata fields: free text attributes and atomic attributes. Free text fields serve as documentation for a case, whereas atomic attributes are descriptors used for indexing (see section 4.2). Our free text metadata consist of the following attributes:

- Goals. Summary of the problem, including the contraints considered.
- Theoretical model. Theoretical basis of the implementation and bibliographic references.
- Study area. Description of the geographic location and the environmental characteristics of the region (e.g. climate, soil, vegetation).
- Input data. Enumeration of the kinds of input data, citing the methodology used in gathering data and the date of collection. Often, maps used in GIS have associated documentation files which already provide this kind of information.
- Outcome. Textual analysis of the effectiveness of the solution represented by the workflow. This is not meant for use by the retrieval mechanism but to help to discriminate between good and bad solutions to a problem. This kind of metadata is very useful, but is seldom, if ever, available.
- Expert or institution. General information about the persons responsible for the model.

The next section discusses the indexing scheme employed.

4.2 Indexing

Stored cases should be indexed to be retrieved efficiently. The same case can be "remembered" in several different ways. Ideally, this requires establishing as many indexing schemes as there are ways in which a case can be remembered, which is obviously unfeasible.

Free-text metadata complicate indexing and require instead the use of Information Retrieval techniques [4] to adequately process the metadata fields of every case. For performance reasons, we restricted our indexing structures to *atomic* metadata fields (named the case *descriptors*). The indexing scheme proposed is based on two kinds of user-provided case descriptors: *goal descriptors* and *constraint descriptors*. Goal descriptors are used to define to what *problem* a case is related, acting as a first filter over the case set, since there can be various solutions related to a given problem. Constraint descriptors differentiate

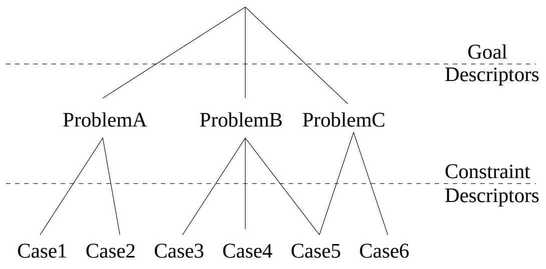


Figure 5: Goal and constraint descriptors partitioning of the set of cases.

among *cases* that treat a problem and represent characteristics specific to this problem. Figure 5 illustrates the organization behind this idea.

Goal descriptors are keyword expressions for a problem statement – e.g. "fertilizer application". Constraint descriptors are predicates composed by triples of the form $< attribute, operator, value >$. The *attribute* field is the name of a well-defined variable/characteristic of a particular problem, and can have string or numeric *values.* The *operator* fiels is a relational comparison operator (e.g. $=$, $>$). For string-valued attributes, the only valid operator is equality ($=$).

An example of a constraint descriptor is $< culture, `=', soybeans > \wedge < soilTexture, `>', 50\% >$. The *operator* and *value* fields differentiate among the existing cases related to a problem. All goal descriptors are unweighted. Constraint descriptors, on the other hand, have relevance weights, set by users.

Case indexing consists in constructing a set of data structures linking these descriptors to a case. First, the *terms* encapsulated in the descriptors (goal descriptors and names of attributes and string values of constraint descriptors) compound a thesaurus, which is augmented by users when necessary. Next, terms are interrelated to represent their semantic context through hierarchical and equivalence links, organized as a forest of n-ary trees, where each node is a term.

The concept of hierarchical (vertical) and equivalence (horizontal) linked structures has been, for instance, used for many years by experts of agriculture to standartize the vocabulary employed in documentation tasks. Examples of agricultural thesauri are AGROVOC [20], developed by the Food and Agriculture Organization of the United Nations, and THESAGRO [11], developed by the Brazilian Ministry of Agriculture. Figure 6 portrays how the term "maize" is represented in AGROVOC. It shows that "maize" is a kind of "cereal", "corn" is a term equivalent to "maize", and "dent maize" and "popcorn" are narrower (more specialized) terms. The notion of thesauri is now being incorporated into the use of ontologies to classify domains. Ontology management is receiving increased acceptance by people working on geographic problems (e.g. [19, 21]). In particular, the OntoWEDSS prototype [10] presents an example of integration of ontologies to case-based and rule-based reasoning for wastewater treatment.

Metadata, descriptors and indexing are the basis for establishing our retrieval scheme, discussed in the next section.

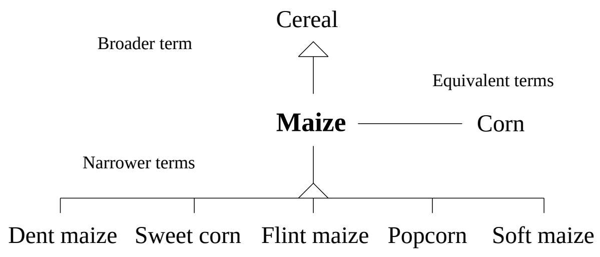


Figure 6: Representation of the term “maize” in AGROVOC.

4.3 Similarity based retrieval and analysis

Similarity retrieval and analysis implies finding, for a given user query, the database objects that are the “most similar” to the user criteria. This is a huge task, because similarity is highly domain dependent. Similarity-based retrieval is an open issue in several domains, such as image databases e.g., [33]. The approach requires defining the notion of “similarity”, by determining similarity functions and associated metrics. One of the problems encountered in this context is that queries are imprecise, since the user has a general idea of desired characteristics, but exact matching is usually impossible.

Santini [42] distinguishes between queries where users know what they are looking for, though without an exact description, and queries where users just have a general idea of what they are looking for, and refine their search criteria interactively. This last kind of search procedure is called feedback similarity based retrieval.

Similarity criteria are usually associated with distance metrics, which are applied to compute the distance between the query goal and what is stored. Stored objects are associated with descriptors that are used to compute the distance between a user request and the retrieved database objects. In our context, cases (< scientific workflow, metadata > pairs) are the stored objects. Case descriptors used for retrieval are part of a case's metadata. Global descriptors are those that describe an entire case, while local descriptors describe parts of a case. The notion of partial and global descriptors is also common in other similarity based retrieval studies - image or multimedia databases, but also textual information retrieval as well. Several mathematical models have been proposed for similarity metrics, some of which started from psychological studies. A taxonomy of such proposals is presented in [24].

The general process of similarity analysis employed in CBR systems can be described as follows [30]:

1. Find correspondences. This step has the intention of aligning the attributes of the input problem and the stored cases to be compared, when they are not directly mapped into each other;

2. Compute the degree of similarity of corresponding features. This step

combines many evaluation metrics to compare corresponding sets of different kinds of attributes; and

3. Assign importance values to features.

This whole process is intrinsically complex. First, finding semantic correspondences requires interpretation of the relationships among features to identify their functional role, which is very costly and knowledge-intensive. Second, the function(s) that compute similarity among features must be tuned according to the domain considered. For instance, two geographic features one meter apart can be very close or very far away, depending on the situation. Finally, it is not easy to quantitatively define the relevance of each problem characteristic, especially for non structured domains, as is our case.

Minkowski's r-metrics are amongst the most common metric-based similarity measurements used in databases:

$$L_r(x,y) = \left[\sum_{i=1}^{n} |x_i - y_i|^r\right]^{1/r}, r \geq 1 \tag{1}$$

$$L_\infty(x,y) = max_i|x_i - y_i| \tag{2}$$

where x, y are multidimensional descriptors of two database objects. For $r = 1$ and $r = 2$ we respectively have the so-called *city-block* and *Euclidean* metrics. These metrics satisfy a set of well known axioms:

$$\begin{array}{ll} \text{Self-similarity:} & d(S_a, S_a) = d(S_b, S_b) \\ \text{Minimality:} & d(S_a, S_a) \leq d(S_a, S_b) \\ \text{Simmetry:} & d(S_a, S_b) = d(S_b, S_a) \\ \text{Triangular inequality:} & d(S_a, S_b) + d(S_b, S_c) \geq d(S_a, S_c) \end{array} \tag{3}$$

where S_a, S_b describe two database objects, and d their distance.

Euclidean metrics, for instance, are often applied to compare two images in terms of color features, since they are appropriate to model the human notion of similarity in this context. The similarity evaluation method employed in WOODSS' CBR mechanism is based on the *nearest-neighbor* idea [30], which uses *city-block* metrics. This method can be expressed by the equation:

$$Similarity(I, S) = \sum_{i=1}^{n} w_i \times f(I_i, S_i) \tag{4}$$

Where:

- I is the input case;
- S is a stored case;
- n is the number of attributes in each case and i is an individual attribute;

- f is a similarity function defined for computing similarity between cases I and S; and
- w_i is the importance weight of attribute i in computing the similarity of I and S.

Similarity values are usually normalized to fall within a range of 0%-100%, where 0% means totally dissimilar and 100% is an exact match. Limitations of this method are that potentially the whole case set needs to be analysed and that it considers only direct correspondences among features. Advantages are that it is relatively simple to implement and it never discards a good match prematurely.

In WOODSS, the user provides parameters for goal and constraint descriptors (input case) and the system returns the cases closest to this input, using the function given in (5).

$$Similarity(I, S) = simGoal(I, S) + simConstr(I, S) \tag{5}$$

Where:

- I is the input case;
- S is a stored case;
- $simGoal(I, S)$ is the similarity value of the goal descriptors of I and S; and
- $simConstr(I, S)$ is the similarity of the constraint descriptors of I and S.

Both *simGoal* and *simConstr* are computed using equation (4).

Similarity analysis is performed in two steps, following the idea presented in figure 5. First, the algorithm identifies the stored problems related to the input case. This is the role of the *simGoal* calculation. *simGoal* uses cases' goal descriptors (unweighted keywords), comparing each goal descriptor of the input case with all of the selected stored cases, and selecting the greatest similarity value for each input goal descriptor. The final *simGoal* value is given by the average of the selected similarity values of the input goal descriptors. The output of this first step are the cases with the largest value for *simGoal*.

The second step evaluates the similarity among the constraints (*simConstr*) of the input case and of the cases identified in the first step. We recall that constraints are predicates composed by conjunctions of triples of the form $< attribute, operator, value >$, where each triple has a particular relevance weight, assigned by the user who created the model. For each triple, the algorithm calculates the similarity between the *value* fields of corresponding (identical) *attribute* names, and *simConstr* is given by the weighted average of these values.

The similarity between two goal descriptors when they are string constraints is based on computing their distance in the thesaurus (the number of edges traversed in the path between them). Zero edges traversed (the node is compared to itself) means an exact match. When the path is not encountered traversing the maximum number of edges stipulated (in the current implementation, five edges), there is a total mismatch.

The similarity between two numeric constraints requires considering their *operator* field (for strings, only the operator '=' is permitted). Numeric constraint operators define intervals, with one side limited by the constraint value field (e.g. $< slope, `<', 0.05 >$ has an upper bound limit of 0.05). When the operator is '=' this interval is a point. The similarity between two constraints is given by the percentage of the interval defined by the input constraint which falls within the interval defined by the stored case's constraint. Thus, if the input interval is completely contained in the stored case's interval, there is an exact match, and when they are disjoint, there is a total mismatch.

4.4 Retainment and retrieval mechanisms

Having defined case indexing and similarity computation, the last task in constructing a CBR mechanism is to define its retainment and retrieval schemas. A retainment mechanism must: (1) decide if a new case is to be retained; (2) request user intervention to provide indexing descriptors; and (3) store the case in the Case Base (WOODSS' modelbase). A new case should only be retained if it is sufficiently "different", in its semantics, from the others in the Case Base. Because of the inherent complexity of automatically taking this decision, user input is required. To help the user in this decision, WOODSS takes the descriptors initially defined for the new case and browses the modelbase, retrieving the most similar cases. These retrieved cases help the user to decide whether to store the new case and also to refine the initially defined descriptors. Each new case is stored in the modelbase in terms of database tuples. Cases are stored only upon user request.

The retrieval mechanism was defined according to a standard procedure. It starts by taking the input problem descriptors (provided by the user) and returning the most similar cases, using the similarity metrics already presented.

4.5 Comparison to other CBR systems

Case based reasoning has traditionally been applied to business situations – e.g., in handling work processes or in the customer service area. This kind of application domain is more structured than environmental applications, and constraints are well defined. In this kind of scenario, the case base is not supposed to grow indefinitely, and thus case storage management is less crucial. We, on the other hand, had to consider the issue of multiple applications and users, in a less structured context, and thus to handle cases within a database management system. Indeed, since most CBR situations deal with specific applications, the focus is on case learning and construction, and thus case representation and storage are often not described, or treated at the memory (algorithmic) level.

Section 2 pointed at examples of the use of CBR in the environmental context – e.g., [7, 46, 23, 3, 37, 25, 10]. Each of these systems is concerned with a specific kind of application, and their algorithms have been optimized to deal with it – they are centered on executing a set of models using different input parameters. We, on the other hand, present a generic framework, where the cases can grow indefinitely. This flexibility was achieved at the cost of several simplifications. One such simplification was to require user intervention at the input or update of every case to provide metadata and descriptors. Another simplification was to limit descriptor indexing to atomic metadata and constraints.

The approach we take – using workflows as a basis to represent cases – is close to the work of Kim et al. [29], in the sense that they also retrieve workflows. However, their study is geared towards business situations. Furthermore, it is strongly based on retrieval by keywords, whereas WOODSS also allows retrieval according to problem constraints.

CBR research – and, more particularly, the work surveyed on environmental CBR systems – is highly influenced by artificial intelligence techniques. It is thus concerned with algorithms that foster case learning (e.g., the neural networks of [12], the graphs of [28]) or the combination of ontologies and rules of [10]. WOODSS instead focuses on robustness of case storage and retrieval, taking advantage of database management techniques. This enhances flexibility in case management, and allows easy updating and extending the casebase. On the other hand, this presents drawbacks in terms of case learning, with a larger burden placed on the user.

5 Example of use

As mentioned in section 3, the first version of WOODSS was centered on database exact match retrieval. A new version of WOODSS was implemented to include the described CBR-based retrieval and storage mechanisms. It uses the Java^{TM} language and is coupled to the Idrisi GIS [14], sold by Clark University. Similar to other CBR-related systems, it needs this specific software to run, since cases are related to planning within Idrisi (e.g., see [28] that is coupled to Microsoft Project). This section presents experiments conducted using CBR through a practical example, from the agro-environmental domain.

5.1 Problem description

The problem analysed in this example is referred to as "variable rate lime and fertilizer recommendation for crops". It corresponds to defining procedures for applying variable rates of nutrients in a given region to optimize crop productivity. Some nutrients in soil moisture are essential to plants (e.g. nitrogen, phosphorus, potassium). To guarantee a good yield these nutrients need to be continuously restored, since they are consumed by crops and lost by erosion. Furthermore, soil acidity reduces the availability of these nutrients and must be neutralized, by improving soil pH. Fertilizers and lime are the substances respectively applied to solve these problems.

This kind of procedure is recurrent in all kinds of crop management situations. Thus, it is a typical case to be stored, since it describes models and activities that need to be performed when one deals with agro-environmental planning. For instance, fertilizers may produce undesirable side effects in the environment (such as pollution of water sources), and thus their use is subject to control. Distinct experts recommend different kinds of products/brands to achieve nutrient balance, and consider various algorithms. Moreover, products and policies are highly crop and region-dependent. Therefore, this problem comprises a large number of cases and is a good example to present in the CBR environmental context.

The decision on amounts of fertilizer to apply is based on soil samples of the focused region as well as on its geographical characteristics (e.g. latitude, rainfall). Lime calculation considers the balance of ions in the soil moisture such as $Al^{3+}, H^{+}, Ca^{2+}, Mg^{2+}$ and K^{2+}. Fertilizer amount calculation identifies the lack of these nutrients, and the estimated response of the culture. Both recommendations vary according to many features, among others: the region, the type of culture, the class of soil, climate, the crop rotation policy and the residual effect of previous lime and fertilizer applications.

The traditional method consists in uniformly applying the same quantity of product - fertilizer or lime - everywhere. In a *precision farming* environment, product application is localized, considering the spatial variability of the features considered, and is specified via maps generated by decision processes. Maps are developed using a GIS and embarked in special agricultural machinery equipped with location sensors (e.g. tractors) which applies the correct amounts of product at each spot.

This example focuses in the first part of this process: the generation of the recommendation maps. A solution process to this problem is composed of two main steps: (a) generation of a map for lime quantity recommendation, and (b) generation of maps of fertilizer quantities, one for each nutrient considered.

5.2 Variable rate lime and fertilizer recommendation using a GIS

(a) Generation of the map of lime quantities. There are many available models to calculate the adequate quantities of lime. Consider that the best model for the region under study is [36]:

$$Lime(ton/hectare) = \frac{CEC \times (V2\% - V\%)}{10 \times RTPN} \tag{6}$$

Where:

- CEC: Cation Exchange Capacity of the soil, given by $CEC = Ca + Mg + K + H + Al\ mmol_c/dm^3$;
- $V\%$: the current Base Saturation of the soil, which expresses the percentage of Ca, Mg, and K ions on the CEC ($V\% = 100 \times \frac{Ca+Mg+K}{CEC}$);

- $V2\%$: the desired percentage of base saturation; and
- $RTPN$: Relative Total Power of Neutralization, which is the expected reaction of the kind of lime employed.

The following algorithm describes an implementation of this equation in the Idrisi GIS. The inputs are maps with the distributions of Ca, Mg, K, H and Al for the region, and the given values of $V2\%$ and $RTPN$.

The following steps are performed in Idrisi:

1. Calculate the CEC, by successively overlaying the maps of Ca, Mg, K, H and Al (*summation* option of the module *overlay*);
2. Compute $V\%$, by obtaining the amount of Ca, Mg, and K ions on the CEC (*division* option of *overlay*) and translating to percentage (*multiplication by value* option of the module *scalar*);
3. Calculate $CEC \times (V2\% - V\%)$, using the *subtraction by value* of *scalar*, and next combine the resulting map with the CEC map (*multiplication* option of *overlay*); and
4. Finally, obtain the lime recommendation map, by adjusting the value calculated in the previous step to the kind of lime used (divide by $RTPN \times 10$ using the *division by value* option of *scalar*).

(b) Generation of maps of fertilizer quantities. This example considers only potassium (K) and phosphorus (P), which are, in conjunction with nitrogen, the nutrients usually most consumed by plants. The fertilizer recommendation is based on tables provided by institutes of agriculture research that relate the availability of the nutrient in the soil with the adequate quantity of product to be applied, according to the culture, climate and soil type considered. The values provided by these tables need to be adjusted to the chosen fertilizer, since different products and brands have distinct concentrations of the nutrient. One algorithm to generate this recommendation in Idrisi, taking as inputs the maps of distribution of K and P, is given by:

1. Process the nutrient maps (K and P) setting the adequate quantities of fertilizer, using values in the respective recommendation tables (*reclass* module); and
2. Adjust the recommendation to the fertilizers to be applied, according to the concentration of the product (*division by value* option of the module *scalar*).

5.3 User interaction with WOODSS

Assume now that a user wants to develop fertilizer and lime recommendation maps for soybean in a region in the Campinas county (Brazil), using WOODSS.

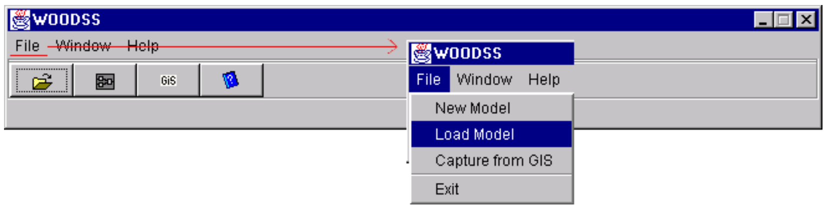
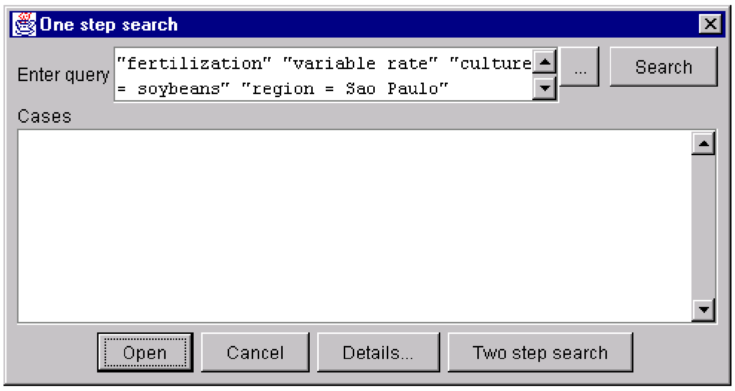


Figure 7: Main window menus in WOODSS.



"fertilization" "variable rate" "culture = soybeans" "region = Sao Paulo"

Figure 8: Search screen.

First the user will want to verify if previous solutions to this problem have been developed. The user browses WOODSS' modelbase to check for similar cases. Figure 7 shows the main menu of the CBR version of WOODSS. By clicking on "Load model" in menu "File" the user launches the search process and provides the problem descriptors. The "Enter query" field in figure 8 shows a screen copy of a query for similar cases, where the goal descriptors entered are "fertilization" and "variable rate" and the (string-valued) constraint descriptors are "culture = soybeans" and "region = Sao Paulo". The system returns the most similar cases reached, using the retrieval scheme described previously. The user has then two possibilities: (a) to select one or more of the returned cases and open them in individual workflow handling windows, or (b) to refine the search.

If one of the cases returned satisfies the user, the user can then execute the corresponding workflow in Idrisi and generate the desired output maps. If the user identifies one case that satisfies only partially his/her needs, the user reuses this case by adding or removing activities and dependencies and adjusting activities' parameters via WOODSS' user interface.

In this specific situation, suppose that no returned case satisfies user needs, and thus the bottom part of the search screen is blank (see figure 8). Here, the user has two alternatives: (1) start the modeling process in the Idrisi GIS, or (2) start the modeling process through the definition of a new workflow in WOODSS. In either case, the model will be represented by a workflow, and

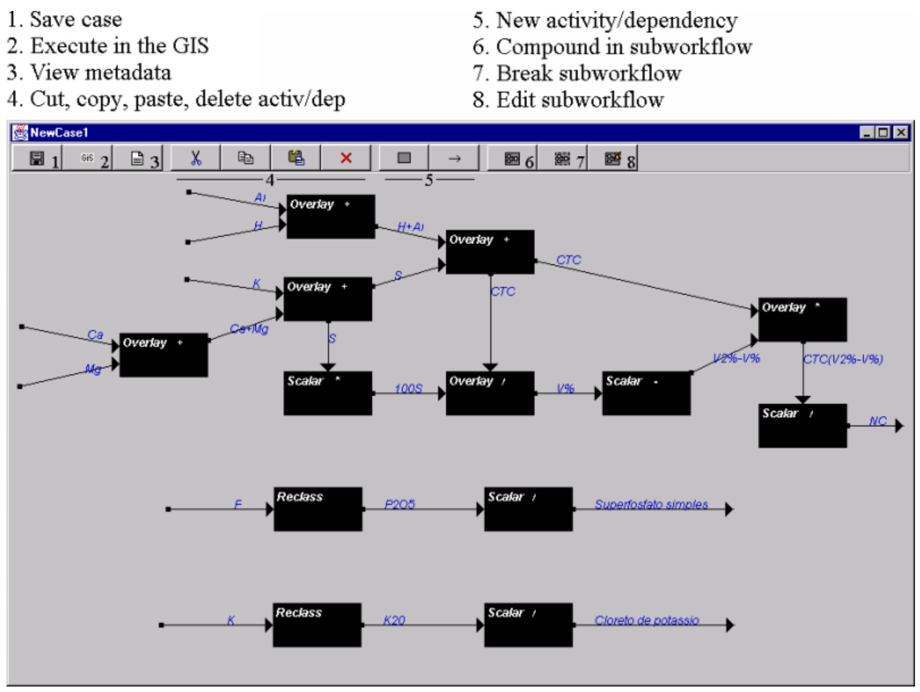
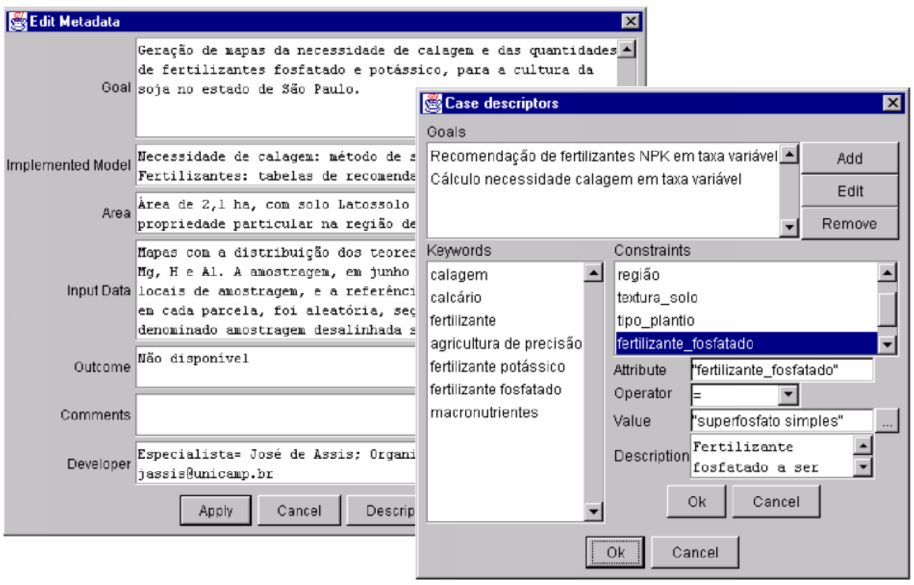


Figure 9: Workflow corresponding to the example.

the case by an instantiation of the workflow together with associated metadata. Figure 9 shows the result of the definition of this workflow.

After finishing the construction of the model, the user may want to store it in the modelbase. By clicking on button "Save case" (figure 9) the system requests the user to fill the metadata fields to be associated with the new case, and to provide the goal and constraint descriptors to index the new case. Figure 10 portrays the forms to define metadata and descriptors for a case.

Considering the given descriptors, the system queries the modelbase and returns the cases most similar to the new one. Next, the user analyses these cases to decide if the new case should be retained or not. We recall that all terms used as descriptors need to be in the thesaurus, thus the user may have to add new terms to it. Thesaurus term entering and querying is done through a tree structure, as shown in figure 11. The basis for the WOODSS thesaurus was the Brazilian Agriculture Ministry Thesagro [11]. This structure permits handling synonims and abstractions of terms, leading to a more context sensitive descriptor comparison.



Goal: Geração de mapas da necessidade de calagem e das quantidades de fertilizantes fosfatado e potássico, para a cultura da soja no estado de São Paulo.

Implemented Model: Necessidade de calagem: método de s
Fertilizantes: tabelas de recomenda

Area: Área de 2,1 ha, com solo Latossolo
propriedade particular na região de

Input Data: Mapas com a distribuição dos teores
Mg, H e Al. A amostragem, em junho
locais de amostragem, e a referênci
em cada parcela, foi aleatória, seg
denominado amostragem desalinhada s

Outcome: Não disponível

Comments:

Developer: Especialista= José de Assis; Organi
jassis@unicamp.br

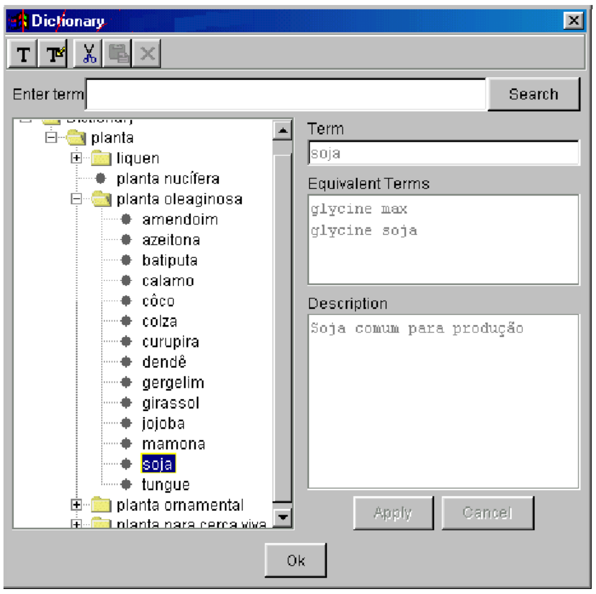
Goals:
Recomendação de fertilizantes NPK em taxa variável
Cálculo necessidade calagem em taxa variável

Keywords:
calagem
calcário
fertilizante
agricultura de precisão
fertilizante potássico
fertilizante fosfatado
macronutrientes

Constraints:
região
textura_solo
tipo_plantio
fertilizante_fosfatado

Attribute: "fertilizante_fosfatado"
Operator: =
Value: "superfosfato simples"
Description: Fertilizante fosfatado a ser

Figure 10: Metadata and indexing descriptors associated to the example.



Enter term:

planta
liquen
planta nucífera
planta oleaginosa
amendoim
azeitona
batiputa
calamo
côco
colza
curupira
dendê
gergelim
girassol
jojoba
mamona
soja
tungue
planta ornamental
planta para cerca viva

Term: soja

Equivalent Terms:
glycine max
glycine soja

Description: Soja comum para produção

Figure 11: Thesaurus manipulation window of WOODSS.

6 Concluding remarks and future work

This paper presented the CBR-centered implementation of a computational tool named WOODSS (WOrkflOw-based spatial Decision Support System), developed at the LIS laboratory of the Institute of Computing of UNICAMP in Brazil. The main goal of this tool is to help decision makers to profit from past problem solving and collaboratively exchange their experience using CBR to lever sharing. This work combines research on database systems, artificial intelligence and workflow management.

WOODSS is centered on dynamically monitoring user activities in a GIS and documenting them using scientific workflows. Besides documenting decision processes on the fly, these workflows can be used to guide experts in solving analogous problems, or as partial solutions to a bigger problem, allowing a global view of the current state of a decision process, and helping to justify decisions.

This paper discussed workflow retrieval and storage mechanisms designed using CBR techniques in order to provide context sensitive analysis. This scheme helps users to identify the most adequate past solutions to support a new decision process, minimizing the reuse/adaptation effort to fit this solution to the new situation. This approach is proving useful to end-users, whose work habits require checking alternative solutions, but who have little support for this kind of verification.

Ongoing work involves various directions. At the moment, we are enhancing WOODSS to allow associating metadata at several levels, to improve retrieval and documentation. We are also working with domain experts to extend the modelbase. Another possible improvement is to provide algorithms to support weight assignment to constraints.

The presented retrieval scheme considers only metadata descriptors. Thus, it must be extended do handle also georeferenced data, using GIS functions in the similarity evaluation. We are now working on coupling domain ontologies to WOODSS, which will also help the vocabulary issues already mentioned [18]. Another extension that might be considered refers to the performance of the nearest neighbor similarity function, which degrades as the number of stored cases increases. Nearest neighbor evaluation would be improved using scalable indexing structures. One alternative structure is the M-tree [13], extensively used in image databases. This structure indexes objects based in their similarity distances in a multidimensional metric space. Thus, its use might prove interesting in WOODSS and other CBR applications.

Finally, the implementation was conducted in a single-user environment. We are now extending it to a multiple-user situation, for workflow design and testing, with support for participatory planning involving distinct experts. In this sense, workflows can act as a communication means, where partial workflows developed by distinct (groups of) users can be combined to compose a cooperative solution. Groups of users can develop partial or alternative solutions at different times, and keep track of what everyone is doing by looking at the respective workflows.

The CBR facilities within WOODSS have been coded in JAVA, and use WOODSS' database (coded in the commercial product FoxPro). It runs coupled to the Idrisi GIS [14], on Windows98. The JAVA code is available upon request. In order to work properly, it needs to be coupled to this GIS and DBMS.

Acknowledgements

This work was developed with financial support from CNPq and from projects SAI – Advanced Information Systems – of PRONEX MCT/FINEP, WebMaps of CNPq, and Interoperability in GIS of CNPq/NSF. The authors thank the domain experts from Faculty of Agriculture Engineering of UNICAMP and the soy bean research center of the Brazilian Agriculture Research Company (EMBRAPA/CNPSoja) for sharing their knowledge.

References

[1] A. Aamodt and E. Plaza. Case-based reasoning: foundational issues, methodological variations and system approaches. *AI Communications*, 7(1):39–59, 1994.

[2] A. Ailamaki, Y. Ioannidis, and M. Livny. Scientific Workflow Management by Database Management. In *Proc. 10th IEEE International Conference on Scientific and Statistical Database Management*, pages 190–201, 1998.

[3] P. Avesani, S. Terral, and E. Martin. Final report. CARICA deliverable V11. Technical report, IRST, 1997. Technical Report.

[4] R. Baeza-Yates and B. Ribeiro-Neto. *Modern Information Retrieval.* Addison-Wesley, Wokingham, 1999.

[5] W. Bausch, C. Pautasso, R. Schaeppi, and G. Alonso. BioOpera: Cluster-Aware Computing. In *Proc. of the 4th IEEE International Conference on Cluster Computing*, 2002.

[6] H. K. Bhargava, S. Sridhar, and C. Herrick. Beyond Spreadsheets: Tools for Building Decision Support Systems. *Computer*, pages 31–39, 1999.

[7] K. L. Branting and J. D. Hastings. An empirical evaluation of model-based case matching and adaptation. In *Case-Based Reasoning Workshop*, pages 72–78, Seattle, Washington, 1994. American Association for Artificial Intelligence.

[8] J. D. Carswell, D. Wilson, and M. Bertolotto. Digital Image Similarity for Geo-spatial Knowledge Management. In Springer, editor, *Proc 6th European Conference on Advanced in Case-Based Reasoning ECCBR 2002*, number 2416 in LNCS, pages 58–69, 2002.

[9] M. Cavalcanti, M. Mattoso, M. Campos, F. Llirbat, and E. Simon. Sharing Scientific Models in Environmental Applications. In *Proc ACM Symposium Applied Computing - SAC*, pages 453–457, 2002.

[10] L. Ceccaroni, U. Cortes, and M. Sánchez-Marré. OntoWEDSS: Augmenting Environmental Decision-support Systems with Ontologies. *Environmental Modelling & Software*, 2004. Article in press, Corrected proof available online 2 dec 2003.

[11] Brazilian Agriculture Ministry CENAGRI. Thesagro – Brazilian National Agricultural Thesaurus, 1997. In Portuguese.

[12] K. Choy, W. Leeb, and V. Lo. Design of an intelligent supplier relationship management system: a hybrid case based neural network approach. *Expert Systems with Applications*, 24(2):225–237, 2003.

[13] P. Ciaccia, M. Patella, and P. Zezula. M-tree: An efficient access method for similarity search in metric spaces. In *Proceedings of the 23rd International Conference on Very Large Databases*, pages 426–435, 1997.

[14] USA Clark Labs/Clark University Worcester, MA. Idrisi. http://www.clarklabs.org, as of Jan 2002.

[15] Federal Geographic Data Committee. Content standard for digital geospatial metadata. http://www.fgdc.gov/metadata/metadata.html, as of Jan 2004.

[16] M. D. Crossland, B. E. Wynne, and W. C. Perkins. Spatial decision support systems: an overview of technology and a test of efficacy. *Decision Support Systems*, 14(3):219–235, July 1995.

[17] C. Ferris and J. Farrell. What are Web Services? *Communications of the ACM*, 46(6):31–35, 2003.

[18] R. Fileto, L. Liu, C. Pu, E. Assad, and C. B. Medeiros. POESIA: An Ontological Workflow Approach for Composing Web Services in Agriculture. *VLDB Journal*, 12(4):352–367, 2003.

[19] F. Fonseca, C. Davis, and G. Camara. Bridging Ontologies and Conceptual Schemas in Geographic Applications Development. *Geoinformatica*, 7(4):355–378, 2003.

[20] Food and Agriculture Organization of the United Nations. AGROVOC multilingual thesaurus. http://www.fao.org/agrovoc/, as of Jan 2004.

[21] R. Frank and Z. Kemp. Ontologies for decision support in environmental information systems. In *Proceedings of GIS Research UK, 9th National Conference, Glamorgan Wales, UK*, pages 53–58, 2001.

[22] D. Hollingsworth. *Workflow Management Coalition - The Workflow Reference Model.* Workflow Management Coalition, 1995. Doc Number WFMC-TC00-1003.

[23] A. Holt and G. L. Benwell. Applying case-based reasoning techniques in gis. *The International Journal of Geographical Information Science*, 13(1):9–25, 1999.

[24] R. Jain, S. Jayaram, and P. Chen. Similarity Measures for Image Databases. In *Proc. SPIE Storage and Retrieval for Image and Video Database III*, volume 5, pages 58–65, 1995.

[25] E. Kalapanidas and N. Avouris. Short-term Air Quality Prediction using a Case-based Classifier. *Environmental Modelling & Software*, 16(3):263–272, 2001.

[26] D. Kaster. Combining Databases and Case Based Reasoning for Decision Support in Environmental Planning. Master's thesis, UNICAMP, december 2001. In Portuguese.

[27] D. S. Kaster, H. V. Rocha, and C. B. Medeiros. Case-based Reasoning Applied to Environmental Modeling with GIS. In *Proc GISCIENCE - First International Conference on Geographical Information Science*, pages 154–155, 2000. Extended abstract.

[28] X. Ke and H. Munoz-Avila. Maintaining Consistency in Project Planning Reuse. In Springer, editor, *Proc. Fifth International Conference on Case-Based Reasoning - ICCBR*, number 2689 in LNAI, pages 665–678, 2003.

[29] J. Kim, W. Suh, and H. Lee. Document-based Workflow Modeling: a Case-based Reasoning Approach. *Expert Systems with Applications*, 23(2):77–93, 2002.

[30] J. L. Kolodner. *Case-Based Reasoning.* Morgan Kaufmann Publishers, San Mateo, CA, 1993.

[31] K. Lancaster, J. Spry, P. Dalton, and M. Shoolbred. Harvesting Knowledge in an Academic Department. Technical report, University of Central England, CIRT, 2003.

[32] G. Laszewski, I. Foster, J. Gawor, P. Lane, N. Rehn, and M. Russell. Designing Grid Based Problem Solving Environments. In *Proc. 34th Hawaii International Conference on System Science*, 2001. Electronic proceedings. paper available at www.hicss.hawaii.edu/HICSS_34.

[33] J. Martinez and N. Mouaddib. Multimedia and Databases - a Survey. *Networking and Information Systems Journal*, 2(1):89–123, 1999.

[34] W. Michener, J. Brunt, J. Helly, T. Kirchner, and S. Stafford. Nongeospatial metadata for the ecological sciences. *Ecological Applications*, 7(1):330–3421, 1997.

[35] S. Mukammalla and H. Munoz-Avila. Case Acquisition in a Project Planning Environment. In *Proc. Sixth European Conference on Case-Based Reasoning ECCBR*, pages 40–51, 2002.

[36] B. Raij, H. Cantarella, J. A. Quaggio, and A. M. Furlani, editors. *Adubation Recommendations for the State of São Paulo (in Portuguese).* Instituto Agronômico de Campinas e Fundação IAC, 2 edition, 1996. Technical report.

[37] F. Ricci, S. Mam, P. Marti, V. Normand, and P. Olmo. Charade: A platform for emergencies management systems. Technical report, IRST, 1994. Technical Report 9404-07.

[38] C. K. Riesbeck and R. C. Schank. *Inside Case-Based Reasoning.* Lawrence Erlbaum Associates, Hillsdale, New Jersey, 1989.

[39] D. Riordan and B. Hansen. A Fuzzy Case-based System for Weather Prediction. *Engineering Intelligent Systems*, 3:139–145, 2002.

[40] A. E. Rizzoli, J. R. Davis, and D. J. Abel. Model and data integration and re-use in environmental decision support systems. *Decision Support Systems*, 24:127–144, October 1998.

[41] A. E. Rizzoli and W. J. Young. Delivering environmental decision support systems: Software tools and techniques. *Environmental Modelling & Software*, 12(2–3):237–249, 1997.

[42] S. Santini and R. Jain. Beyond Query by Example. In *Proc Sixth ACM Intern, Multimedia Conference*, pages 345–350, 1998.

[43] R. C. Schank. *Dynamic Memory: A Theory of Learning in Computers and People.* Cambridge University Press, 1982.

[44] L Seffino, C. B. Medeiros, J. Rocha, and B. Yi. WOODSS - A Spatial Decision Support System based on Workflows. *Decision Support Systems*, 27(1-2):105–123, 1999.

[45] L. T. Steyaert. A perspective on the state of environmental simulation modeling. In M. F. Goodchild, B. O. Parks, and L. T. Steyaert, editors, *Environmental Modeling with GIS*, chapter 3, pages 16–30. Oxford Univesity Press, New York, NY, 1993.

[46] F. Verdenius and J. Broeze. Generalized and instance-specific modelling for biological systems. *Environmental Modelling & Software*, 14:339–348, 1999.

[47] J. Wainer, M. Weske, G. Vossen, and C. B. Medeiros. Scientific Workflow Systems. In *Proc. of the NSF Workshop on Workflow and Process Automation Information Systems*, 1996.

[48] I. Watson. Case-Based Reasoning and Knowledge Management: a Perfect Match? In *Proc. Florida AI Research Society Conference FLAIRS*, pages 118–123. AAAI Press, 2001.

[49] I. D. Watson. *Applying Case-Based Reasoning: Techniques for Enterprise Systems*. Morgan Kaufmann, 1997.

[50] D. Wilson, M. Bertolotto, and E. McLoughlin. Knowledge Capture and Reuse for Geo-spatial Imagery Tasks. In Springer, editor, *Proc Fifth International Conference Case-Based Reasoning ICCBR*, number 2689 in LNAI, pages 622–636, 2003.