An information states blackboard as an intelligent querying interface for snow and avalanche data.

Epaminondas Kapetanios[1]
M.C. Norrie
A. Frei
Swiss Federal Institute of Technology (ETH), Institute for Information Systems, Zurich, Switzerland

[1]Epaminondas Kapetanios now works in the Harrow School of Computer Science, University of Westminster

An Information States Blackboard as an Intelligent Querying Interface for Snow and Avalanche Data

E. Kapetanios, M. C. Norrie, A. Frei
Swiss Federal Institute of Technology (ETH)
Institute for Information Systems
ETH-Zentrum, CH-8092 Zurich, Switzerland
email:{kapetanios, norrie}@inf.ethz.ch

Abstract

We present the graph-based querying paradigm used in the Regional Avalanche Information and Forecasting System (RAIFoS) for the collection and analysis of snow and weather related physical parameters in the Swiss Alps. The querying paradigm relies upon the issue of interactively constructing a semantically valid query graph on an Information States Blackboard as guided by meta-data elements standing for interpretations of conceptual model, data values and/or operations. The meta-data elements constitute the terms of a meta-data-driven query language (MDDQL) the interpretation of which is done interactively relying on a kind of finite state automaton.

Keywords: *Scientific Information Systems, Meta-data, Visual Query languages.*

1 Introduction

The **R**egional **A**valanche **I**nformation and **Fo**recasting **S**ystem - RAIFoS - is being developed to meet the requirements of querying snow and weather related measurement data as collected by various measurement stations and observers in the Swiss Alps. Measurement stations belong to different measurement networks and refer to geo-locations. 50-60 different physical parameters or variables are being measured and these are assigned particular measurement units. All data are stored in operational databases built upon relational engines and/or file systems. For the sake of convenience and efficiency, most of the non-numerical data are encoded. This leads to the mapping of values expressed by natural language words into, usually, arithmetic symbols.

Currently, the analysis and/or visualisation of this data requires scientific users to pose queries in terms of a data model, schema and value encodings which may be unfamiliar. Moreover, they need to learn the syntax of a database specific query language, usually SQL. Thus, the requirements for operating on data in a semantically correct way involve

- learning the query language syntax
- understanding the underlying data schema in terms of interpretations of constructs such as relations, classes, collections, associations holding among them, and attributes
- the interpretation of value encodings as well as measurement units in which domain values are expressed.

To eliminate these, we elaborated a semantically enriched query answering system (SAQAS) for posing queries to scientific data. Instead of trying to formulate a query in a particular database query language with interpretations based on assumptions about the underlying database schema and data values, the query answering system helps in constructing a semantically enhanced query. At the core of SAQAS is a Meta-Data-Driven Query Language (MDDQL) the terms of which are natural language terms and relate to the implementation symbols used for the realisation of a database schema and/or encodings of data values. Thus, the constructed query can be expressed in terms of natural language elements. However, we also elaborated a graphical query interface such that the construction of queries becomes a matter of navigation through information states on a blackboard expressed by a visual graph-based formalism indicating the connections among MDDQL terms.

The terms of MDDQL are basically categorised as `Concepts`, `Properties`, `Concrete Value Domains` and `Measurement Units`. Currently, the construction of a query is enforced by the ordering of mappings $Concepts \rightarrow Properties \rightarrow ConcreteValueDomains \rightarrow MeasurementUnits$. This means that the user has to begin with an initial set of concepts and keep on refining and/or restricting the selected concepts by extensions of the query graph. All terms of MDDQL constitute a conceptual base configured as a meta-data server and implemented by OMS Java, an object-oriented database system with semantically expressive role and association modelling [20, 16]. The connectionism model underlying the structure of the conceptual or ontological base relies on a multi-layered graph formalism with partially allowed cycles. A transformation component for the mapping of the graph query to a target database specific query language such as SQL has been implemented for the generation of AND-connected (conjunctive) queries.

In this paper, we focus on the conceptualisation and implementation issues underlying the querying interface. Section 2 gives an overview of the related work. Section 3 presents the querying paradigm by illustrating the method of constructing an intended query. Section 4 covers the implementation issues and a discussion of future work as well as a comparison with related approaches. Concluding remarks are given in section 5.

2 Related work

The importance of avoiding an underlying query language formalism when end-users need to pose queries to a database system has received much attention during the last 10-15 years within the database research community and many `Visual Query Systems` (VQS) and/or `Visual Query Languages` (VQL) have been developed to alleviate the end-users' tasks. To provide guidelines for the design of these new interfaces, many proposed visual query languages (VQLs) are analysed and compared in [15] from the user's viewpoint. The analysed features include ER concepts, interaction modes, model diagram transformations, condition specification, aggregate functions, and output organisation.

VQSs are query systems for databases that use visual representations to depict the domain of interest and express related requests [27]. VQSs can be seen as an evolution of query languages adopted in database management systems to improve the effectiveness of Human-Computer Interaction (HCI). Thus, their most important features are those that determine the nature of the human-computer dialogue, in order to maximise user task performance [13]. They mainly rely on the integration of the data model and query language in a user-database interface [15] as well as presentation and interaction components that together form a graphical user interface [19].

Query languages based on visual formalisms have also been proposed [4, 7, 22, 29, 17, 6, 14, 2, 21, 26] in order to tackle the problem of query construction. In all these approaches, the conceptual [26] or implementation model is mainly integrated into the query paradigm. [21] gives an overview of example-based graphical database query languages, in terms of their different capabilities and expressive powers. Most example-based languages are based on revised versions of Codd's domain relational calculus.

Graph based visual formalisms have been addressed for both Object-Oriented DBMSs where traversal paths can be expressed as queries [31, 5], and for a Web query system [30] where a visual user interface, WebIFQ (Web In-Frame-Query), is used to assist users in specifying queries and visualising query criteria including document meta-data, structures, and linkage information. Traversal like approaches using graph queries also underly the development of query interfaces for large clinical databases [23, 24, 10, 3], where ad hoc queries to clinical databases can be developed by a query generator. The query generators mostly use an object-oriented data model which is visualised by directed graphs.

In [28], an approach for multi-paradigmatic visual access to databases is described, which is proposed to achieve seamless integration of different interaction paradigms. The user is provided with an adaptive interface augmented by a user model, supporting different visual representations of both data and queries. The visual representations are characterised on the basis of the chosen visual formalisms, namely forms, diagrams, and icons. To access different databases, a unified data model, the Graph Model, is used as a common underlying formalism to which databases, expressed in the most popular data models, can be mapped.

Graph models have also been used in order to enable the activation of statistical operations within a more flexible data model. The *Vista* visual language proposed in [17] facilitates access by statistical users when they interact with a statistical database in order to directly manipulate data. Despite the fact that a directed acyclic graph has been used as an internal model (Grass) for the database schema representation, the model presupposes a full understanding of the underlying data. Thus there is no consideration of interpretation of data and analysis operations during the query construction.

In the following section, we illustrate the method of how to construct a query with a graph based visual formalism within SAQAS. Since query construction relies upon the conceptual base of terms for MDDQL, syntactic issues are covered by the connectionism model whereas semantic issues are mainly covered by the classification structure of terms as well as by constraints applied to the connections of terms. In section 4, we discuss the similarities and differences between our approach and the related approaches mentioned above.

3 SAQAS querying paradigm

Before describing how a potential query might be constructed through the querying interface, we provide an overview of the model underlying the conceptual or ontological base [25, 12, 11, 1, 18] with which the terms of MDDQL are organised. The presentation is associated with the model's concepts and a set of tasks that can be performed at the user interface. In accordance with the framework for visual interfaces to databases specified in [19], we consider the conceptual base as a database and its environment as specified by the triple $< model, presentation, tasks >$.

The model: The terms of MDDQL are mainly categorised as those belonging to the `Domain of Interest` and those which form `Operations`. The domain of interest is classified as `Concepts`, `Properties`, `Concrete or Closed Value Domains` and `Measurement Units`, which are represented by the sets C, P, CVD, MU, respectively. The connectionism model relies upon binary associations (links) which hold either over the same set of terms (recursive links) to form more complex terms, or among different sets of terms (interconnecting links). `Relationships` are special sorts of *Concepts*. Links are classified as `assertional` - empirically defined - or `definitional` - given by definition. Furthermore, recursive links, are classified as `natural kinds` indicating an IS-A hierarchy, or `compositional` indicating a *composed object*.

For example, when referring to the avalanche research domain, the terms `measurements`, `Observer's data`, `snow data`, `measured by`, `delivers`, `measurement station`, `constitute`, `measurement network`, are all concepts. The terms `temperature`, `height` and `direction` are properties and the terms `North`, `North-East`, `North-West` or $[0, 300]$, are considered as concrete value domains. Note that terms are actually objects assigned unique identifiers which are, in turn, assigned to natural language words (see also section 4). However, for the sake of simplicity, throughout the paper we refer to terms by using the natural language words directly. Generally, this is possible in the case that there are no possible ambiguities for the presentation of terms.

The presentation: The querying paradigm is depicted in figure 1 which illustrates the presentation method of the MDDQL terms. Regardless of how they are classified in the conceptual base, all terms are represented as nodes (rectangles) with the corresponding natural language word as a label. All activated nodes are coloured differently in order to indicate the constructed or selected query. For example, a lighter shaded node in figure 1 indicates a node that has been selected and the darker shaded nodes indicates terms that are available for selection. Currently, we are working on different presentation methods in order to distinguish among concepts, relationships, properties, concrete value domains and measurement units. The terms are connected to each other by lines indicating links. The links can be labelled or coloured differently indicating their role as described above.

The tasks: Currently, the main tasks to be performed by the user on the query interface are: a) the `initialisation of the information states space` by selecting the initial *concepts* of interest which form the starting point for

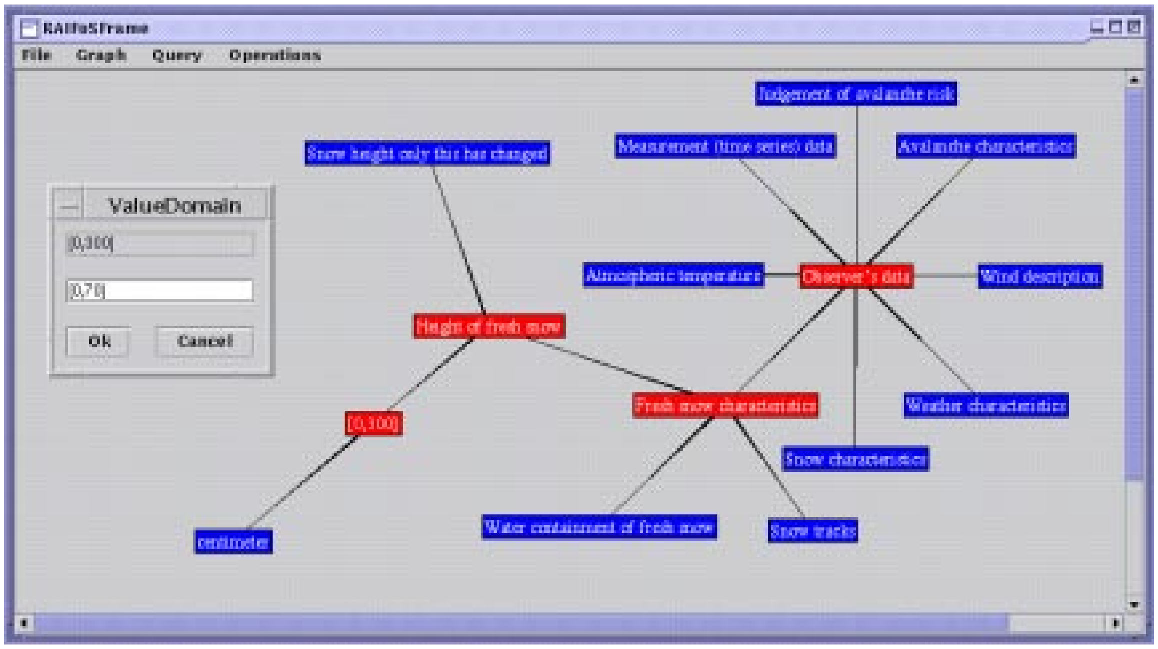


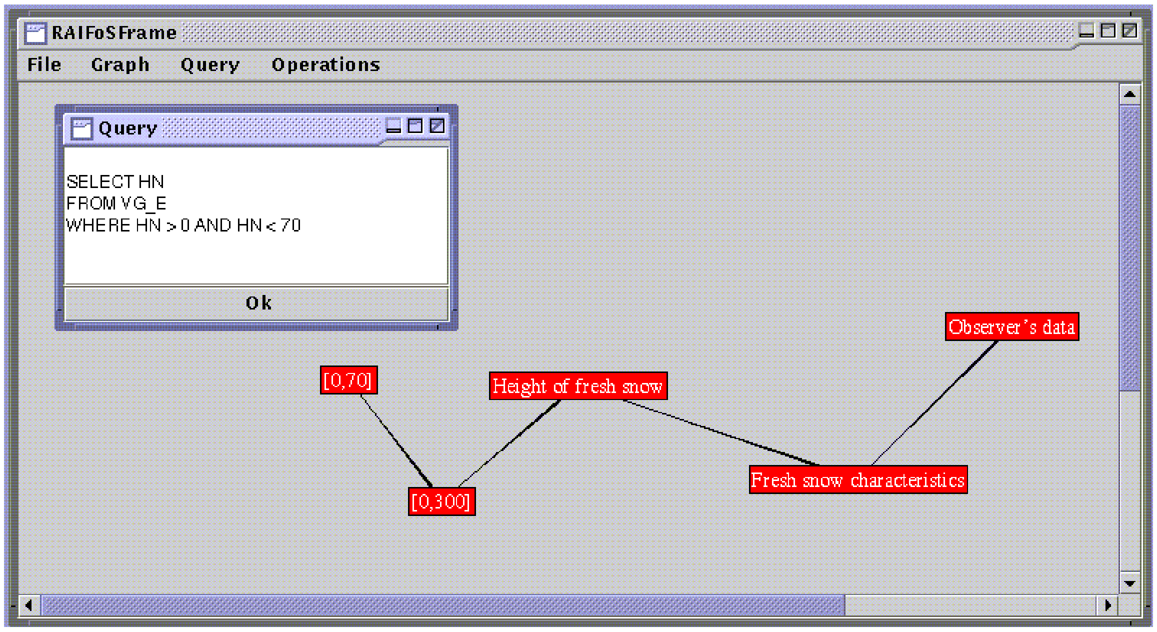
Figure 1. A particular information state

query construction, b) the `activation of a particular node` to be considered in the constructed query, c) `assignment of an operator` to the activated node, d) the `deactivation of a particular node` by removing it from the blackboard, and finally e) the `activation of the answer and/or transformation of the graph query`.

Figure 1 depicts a particular query construction state after having activated the node `Measurement data` and focussing on `Observer's data` which was presented as a kind of *Measurement data* - a link classified as *natural kind* and *assertional*, i.e. an IS-A relationship which is empirically defined. By activating the node for the term `Observer's data`, we receive linked terms associated with this particular term such as the term `measured by` indicating *relationships* - a special sort of *concept* - to other terms (concepts), as well as linked terms standing for characteristic properties such as `Fresh snow characteristics`, `wind description`, etc. If the user would have decided to activate the node (term) *measured by*, this would have led to the linked term *measurement station*, i.e. the user would be interested in *observer's data measured by concrete measurement stations.*

Having activated the node `Fresh snow characteristics` means that the user would like to focus on particular properties associated with the concept `observer's data`. This will cause a move to an adjacent state extending the previous one by drawing links to further properties which might constitute a complex one. In this case, links will be drawn to nodes (terms) standing for further properties such as `Height of fresh snow`, `Snow tracks` and `Water containment of fresh snow`. By activating the node (term) `Height of fresh snow` links will be drawn to the concrete value domain $[0, 300]$ expressed in *cm* which is given by definition. This means that it only makes sense to consider a value within this arithmetic interval when specifying restrictions for this particular property.

Given that this is not an enumerable value domain, activating this node pops up a value specification window through which a particular value range might be given, e.g. $[0, 70]$. For concrete value domains with a small number of elements, a selection menu might be considered. The equivalent expression of the constructed graph query in natural language would be `the height of fresh snow, where height is between` $[0, 70]cm$`, as properties of observer's data which is a kind of measurement data`. An equivalent



```
SELECT HN
FROM VG_E
WHERE HN > 0 AND HN < 70
```

Figure 2. Final graph query and its SQL counterpart

SQL query against the relational database storing measurement data could then be generated as depicted in figure 2.

If more than one property had been selected, then the AND logical operator would be assigned by default. For example, if the node (term) `water containment of fresh snow` had also been selected, the equivalent expression of the constructed graph query in natural language would be `the height AND water containment of fresh snow, where height is between` $[0, 70]cm$`, out of observer's data as a kind of measurements`. Therefore, any constructed queries within which no other logical operators are bound to *property nodes* are interpreted as conjunctive, i.e. AND-connected, queries.

Comparison operators can be assigned to nodes (terms) standing for concrete value domains depending on the nature of values under consideration, for example, categorical values such as `North`, `North-East`, `East` or arithmetic values such as $[0, 300]$. Assignments of particular operators and/or statistical operations to nodes are decided and/or inferred by the *MDDQL Interpreter* by calling appropriate methods of whenever a node is activated. Details of the underlying mechanism are described below.

4 Implementation issues

Figure 3 gives a general overview of the semantically augmented query answering system SAQAS. The `Information States Blackboard` provides the query interface which is used for the construction of the graph-based query. Each time a task is performed on the blackboard concerning the query construction such as *activation of initial states*, *activation of a particular node*, *value specification* or *operator assignment*, the functions provided by the MDDQL Interpreter are called from the blackboard. These have been implemented as object methods which, in the case of the activation of initial states, return the corresponding terms classified as concepts. In the event of node activation, all next states connected with the activated node under consideration of semantic constraints are returned. Finally, the constructed query can be transformed by the `transformation component` into the target database query language such as SQL.

All three components, the Information States Blackboard, the MDDQL Interpreter and the MDDQL transformation component, are implemented in Java by using the Swing libraries [9, 8], and therefore, all components needed for the query construction can be downloaded and run locally, at the *client site*. Actually, they reside on

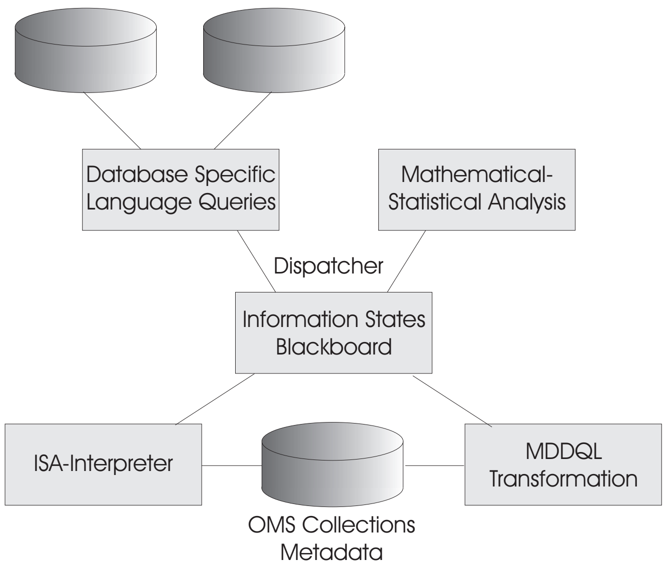

Database Specific Language Queries
Mathematical-Statistical Analysis
Dispatcher
Information States Blackboard
ISA-Interpreter
MDDQL Transformation
OMS Collections Metadata

Figure 3. Architecture of the semantically augmented query answering system

the *application server site* and are down-loaded when a client request arrives. Given the Java-based implementation, it is possible to use the query interface from a Web browser as an *Applet*. The MDDQL ISA-Interpreter realising an Information States Automaton - in the following called simply Interpreter - infers *semantically valid states to move on* and/or *operators* to be considered due to the current query construction state, using the conceptual or ontological base providing the terms of MDDQL. This meta-data resides on the application server site in an OMS Java database and is copied into the workspace of the client at start-up time. This means that no network communication overhead is incurred between conceptual base and MDDQL Interpreter during the interactive query construction process.

Thus, communication with the application server takes place only for the initial downloading of the required components, i.e. interpreter, transformer, conceptual base, and for the transmission of the generated query to the server in order to be further delegated to the target data repository to be executed at its site. It could also be possible to store certain previous and intermediate results — especially in the form of calculated summaries or statistical evaluations — in the meta-data server to avoid re-evaluation of expensive operations on the target data. This feature of the system is currently under development.

Ontological Base and the Meta-Data Server: An MDDQL `term` is conceived as a *unit of thought* and is represented as an object with an object identifier. For example, the terms
$< OM270, Measurement\ data >$,
$< OM260, Snow\ data >$,
$< OM261, Fresh\ snow >$,
$< OM290, Temperature >$,
$< OM330, Height >$
are expressed in English and represented as objects with unique identifiers starting with the prefix OM. A term might be associated with one or more than one word as stated before.

This separation among terms, words (signs) and implementation symbols into which terms of MDDQL might be mapped enables the usage of any kind of natural language without having an impact on the encodings used by the target data repository for database schemas and data values. Additionally, the consideration of terms of the MDDQL as *objects* allows the capture of additional semantic information through *attributes of terms*. For example, a `description` attribute might capture the intensional meaning of a particular term and can also be presented on the blackboard, accompanying the presentation of a graph query node, when needed for further explanation of the node (term).

All terms are objects belonging to the main collection `Alphabet`. The collection *Alphabet* is further classified as `Domain of Interest` and `Operations`. Both are further classified indicating the `role of MDDQL terms within the constructed query` (see also model description in section 3). Construction of a query is done by checking the connectionism model underlying the structure of the conceptual base. The connectionism model is mainly conceived as a *directed acyclic graph* with allowed cycles in particular cases. This enables an $N : M$ assignment of terms within the conceptual base for a particular domain of interest. Thus it is possible to assign, for example, the same primitive concept `wind data` to more than one complex concept (multiple inheritance within IS-A hierarchies) such as `weather data` or `measurement data`, as well as the assignment of the same property to more than one concept. Accordingly, a property may be assigned to more than one concrete value domain.

Further roles may be assigned to the terms such as `primitive` and `complex` properties or concepts, and `categorical` and `numerical` properties. This leads to a rich classification scheme with flexible modelling issues. Links are also classified into various categories such as `assertional`, `definitional`, `natural kinds` and `compositional` and therefore, they must underly the same classification constructs as single objects.

As stated above, the interpreter of MDDQL infers new states to move on and/or operators in conjunction with the structural model of the conceptual or ontological base as briefly described in section 3. Given a particular activated node and the current query construction state, the interpreter infers a subsequent state on the basis of the connectionism model as well as the associated operators. The new inferred state is a set of objects structured by `term unique identifier`, `the assigned word` with which the node will be labelled, a further `description` of the term, its `role` as given by its classification as concept, relationship, property, enumerable concrete value domain, arithmetic interval, etc., as well as the `kinds of links` to the activated node (term), as provided by the representational issues of the conceptual or ontological base.

The structural model for the MDDQL terms is implemented by OMS Java, a Java-based realisation of the OM modelling constructs, which provides rich classification structures and/or constraints for both unary and binary collections of objects. The latter stand for binary associations between objects and these are used to implement the links of the MDDQL connectionism model. A subset of the schema definition language follows indicating the classification and typing issues of terms standing for concepts and properties.

type intdomain
(name : string;
description : string)
type property **subtype of** intdomain
(description : set of string;
dimensionality : integer)
collection DomainOfInterest: **set of** intdomain;
collection Concepts: **set of** intdomain;
collection Entity: **set of** intdomain;
collection Relationships: **set of** intdomain;
collection Properties: **set of** property;
collection Numerical: **set of** property;
collection Categorical: **set of** property;
collection Continuous: **set of** property;
collection Discrete: **set of** property;
constraint Concepts
subcollection of DomainOfInterest;
constraint Properties
subcollection of DomainOfInterest;
constraint MeasureUnits
subcollection of DomainOfInterest;
constraint Numerical
subcollection of Properties;
constraint NaturalKindsOfConcepts
association from
Concepts (0:*) **to** Concepts (0:*);
constraint CharacterisedBy
association from
Concepts (0:*) **to** Properties (0:*);
constraint NaturalKindsOfProperties
association from
Properties (0:*) **to** Properties (0:*);
constraint ConstrainedBy
association from
Properties (0:*) **to** ValueDomains (0:*);
constraint Expressed_in
association from
ValueDomains (0:*) **to** MeasureUnits (0:*);

Managing the elements with a database management system facilitates the maintenance of the conceptual base that is the MDDQL terms repository in a multi-user environment. Changes to individual terms, and consequently, modifications/extensions of the MDDQL alphabet (querying vocabulary) as well as to the structural model itself are due to: a) user interaction - new terms might be specified over already existing ones (top-down approach), and, b) data analysis issues (bottom-up approach). Since MDDQL terms are separated from the MDDQL Interpreter, changes to the terms and/or querying vocabulary have no effects on the application logic of the MDDQL interpreter (syntax and semantics in a querying language uniformly integrated).

Similarities and differences: The approach presented in this paper can clearly be classified as a *graph based query facility or language*, a kind of visual query language which has the advantage over *forms style interfaces* in that they can directly represent relationships within the structure of a query. Given that the presented graph nodes are inferred by the MDDQL interpreter acting as a state automaton in interaction with the user, no learning of a particular syntax of a textual query language is required as happens to be the case in similar query paradigms such as the current graphical query languages `Quiver` [5] and `GOQL` which are compliant to ODMG object databases.

On the other hand, although we also provide a navigational style of traversing relevant information as is the case with some object-oriented and web-based query languages, a more semantically enhanced and restricted navigational style has been adopted with the embedding of interpretations for database schemas, data values and operations as well as constraints holding among them. Since terms of MDDQL are conceived as objects encapsulating more information about their roles in a particular domain of interest, additional icons and/or descriptions might be viewed at the presentation level of MDDQL. For example, it would be possible to give an insight into the intensional meaning of a particular node (term) standing for a property or concrete value by focussing on a node on the blackboard. Alternatively, terms standing for values might be presented iconically where this makes sense.

In contrast with most visual query approaches which are bound to a given conceptual model - using attributes from a particular conceptual schema - we rely on a structural model which enables the expression and/or definition of more complex terms by the user. For example, given that `measurement stations deliver snow related data`, a user might wish to introduce the term `measured snow related data` which will be resolved by the system into the terms `measurement stations, deliver, snow related data`.

Constraints are mainly expressed explicitly by the connectionism model and, therefore, are taken into account when inferences of the MDDQL interpreter are made in terms of establishing or drawing links from an activated node. They are not only restricted to interconnections among concepts, properties, concrete value domains, measurement units and their context based validity, but also to operators which can be bound to particular nodes (terms) due to the current state of query construction. This is particularly useful when statistical or analytical operations are considered.

However, since cycles are allowed in the graph model underlying the ontological base, the consideration of bi-directional relationships during query construction is enabled, such as the equivalence between `measurement data measured by measurement stations` and `measurement stations delivering measurement data` - passive and active cases, respectively, when query construction entry is either the concept `measurements` or the concept `measurement stations`.

Given the distinction between `terms` and `words` and their relationship with the `implementation symbols` (the semiotics triangle) in the underlying the model of the conceptual base of the organisation and management of MDDQL terms, we can enable a switching mechanism among different natural languages such that a query can be constructed on the blackboard by using nodes which are labelled in different languages, e.g. English, German and French.

5 Conclusion

We presented a graphical query interface for a graph based query language MDDQL the terms of which are meta-data standing for the interpretation of implementation symbols such as database schemas, data values encodings and/or operations. An application paradigm referring to a scientific information system for snow and avalanche research data has been used for the illustration of our approach. The query interface is conceived as an information states blackboard where a graph query is constructed interactively and incrementally in conjunction with the MDDQL interpreter. The user can start with an initial concept, and subsequently, activate a particular node in order to get all relevant, linked terms. The linked terms are inferred due to the structural model of the conceptual or ontological base organising the MDDQL terms as well as semantic information and/or constraints embedded within the model.

Currently, the blackboard is being extended to support the tasks of focussing on the textual description (intensional semantics) and/or iconic presentation of the MDDQL terms. Moreover, different presentation schemes should emphasize the classification of the MDDQL terms as *concepts, properties, concrete value domains* and *measurement units*. Links will be described according to their roles such as *assertional* versus *definitional* or *natural kinds* versus *compositional*. Probabilities might also be presented on assertional links.

In addition, the OR and NOT logical operators will be considered in such a way that not only conjunctive queries can be constructed which is currently the case. Furthermore, analysis operations might also be bound to the constructed graph query. They mainly refer to statistical analysis operations as provided by statistical software packages such as `S-Plus`. These operations are also bound to the current state of the query construction due to the inferences made by the MDDQL interpreter. Finally, further considerations of the presentation of the MDDQL query results need to be addressed, since the type of the result is not only restricted to arithmetic or textual values but also to diagrammatic presentations such as pie charts, distribution diagrams and scatter diagrams.

References

[1] Y. Arens, C. Y. Chee, C.-N. Hsu, and C. Knoblock. Retrieving and Integrating Data from Multiple Information Sources. *Int. Journal of Intelligent and Cooperative Information Systems*, 2(2):127–158, 1993.

[2] Berztiss AT. The query language vizla. *IEEE TRANSACTIONS ON KNOWLEDGE AND DATA ENGINEERING*, 5(5):813–825, Oct 1993.

[3] Klaeren H Banhart F. A graphical query generator for clinical research databases. *METHODS OF INFORMATION IN MEDICINE*, 34(4):328–339, Sep 1995.

[4] Santucci G Cardiff J, Catarci T. Semantic query processing in the VENUS environment. *INTERNATIONAL JOURNAL OF COOPERATIVE INFORMATION SYSTEMS*, 6(2):151–192, June 1997.

[5] Manoj Chavda and Peter Wood. Towards an ODMG-Compliant Visual Object Query Language. In M. Jarke, M. Carey, K. Dittrich, F. Lochovsky, P. Loucopoulos, and M. Jeusfeld, editors, *Proc. of 23rd International Conference on Very Large Data Bases*, pages 456–465, 1997.

[6] Wu CT Clark GJ. Dfql - data-flow query language for relational databases. *INFORMATION AND MANAGEMENT*, 27(1):1–15, July 1994.

[7] Florescu D, Raschid L, and Valduriez P. A methodology for query reformulation in CIS using semantic knowledge. *INTERNATIONAL JOURNAL OF COOPERATIVE INFORMATION SYSTEMS*, 5(4):431–467, Dec 1996.

[8] Robert Eckstein, Marc Loy, and Dave Wood. *Java Swing (Swing 1.2)*. O'Reilly and Associates, 1998.

[9] David Flanagan. *Java in a Nutshell*. O'Reilly and Associates, second edition, 1997.

[10] Hripcsak G, Allen B, Cimino JJ, and Lee R. Access to data: Comparing AccessMed with query by review. *JOURNAL OF THE AMERICAN MEDICAL INFORMATICS ASSOCIATION*, 3(4):288–299, July-August 1996.

[11] C. A. Goble, P.J. Crowther, and W. D. Solomon. A Medical Terminology Server. In *Proc. of the 5th Intern. Conf. on Database and Expert System Applications (DEXA 94)*, Lecture Notes on Computer Science, pages 661–670. Springer Verlag, 1994.

[12] N. Guarino and P. Giaretta. Ontologies and Knowledge Bases: Towards a Terminological Clarification. In N.J.I. Mars,

editor, *Proc. of the 2nd Intern. Conf on Building and Sharing Very Large Knowledge Bases.* IOS Press, Enschede, The Netherlands, 1995.

[13] Chan H, Siau K, and Wei KK. The effect of data model, system and task characteristics on user query performance - an empirical study. *DATA BASE FOR ADVANCES IN INFORMATION SYSTEMS*, 29(1):31–49, Winter 1998.

[14] D. Haw, C. Goble, and A. Rector. GUIDANCE: Making it Easy for the User to be an Expert. In P. Sawyer, editor, *Proc. 2nd Int. Workshop On Interfaces to Database Systems*, pages 19–43. Springer Verlag, 1994.

[15] Chan HC. Visual query languages for entity relationship model databases. *JOURNAL OF NETWORK AND COMPUTER APPLICATIONS*, 20(2):203–221, April 1997.

[16] A. Kobler, M. C. Norrie, and A. Würgler. OMS Approach to Database Development through Rapid Prototyping. In *Proc. 8th Workshop on Information Technologies and Systems (WITS'98)*, Helsinki, Finland, December 1998.

[17] Meoevoli L, Rafanelli M, and Ricci FL. An interface for the direct manipulation of statistical-data. *JOURNAL OF VISUAL LANGUAGES AND COMPUTING*, 5(2):175–202, June 1994.

[18] D. Lenat and R.V. Guha. *Building Large Knowledge–Based Systems: Representation and Inference in the Cyc Project.* Addison–Wesley, Reading, MA, 1990.

[19] Murray N, Goble C, and Paton NW. A framework for describing visual interfaces to databases. *JOURNAL OF VISUAL LANGUAGES AND COMPUTING*, 9(4):429–456, August 1998.

[20] M. C. Norrie and A. Würgler. OMS Object-Oriented Data Management System. Technical report, Institute for Information Systems, ETH Zurich, CH-8092 Zurich, Switzerland, 1997.

[21] Wang HQ Ozsoyoglou G. Example-based graphical database query languages. *COMPUTER*, 26(5):25–38, May 1993.

[22] A. Papantonakis and P.J.H. King. Syntax and Semantics of Gql, a Graphical Query Language. *Journal of Visual Languages and Computing*, 6:3–25, 1995.

[23] Liepins PJ, Curran KM, Renshaw CR, and Maisey MN. A browser based image bank, useful tool or expensive toy? *MEDICAL INFORMATICS*, 23(3):199–206, July 1998.

[24] Taira RK, Johnson DB, Bhushan V, Rivera M, Wong C, Huang LJ, Aberle DR, Greaves M, and Goldin JG. A concept-based retrieval system for thoracic radiology. *JOURNAL OF DIGITAL IMAGING*, 9(1):25–36, Feb 1996.

[25] Borgo S., Guarino N., Masolo C., and Vetere G. Using a Large Linguistic Ontology for Internet-Based Retrieval of Object-Oriented Components. In *Proc. of 9th Int. Conf. on Software Engineering and Knowledge Engineering (SEKE 97)*, Madrid, Spain, 1997.

[26] Chan HC Siau KL and Tan KP. Visual knowledge query language. *IEICE TRANSACTIONS ON INFORMATION AND SYSTEMS*, E75D(5):697–703, Sep 1992.

[27] Catarci T, Costabile MF, Levialdi S, and Batini C. Visual query systems for databases: A survey. *JOURNAL OF VISUAL LANGUAGES AND COMPUTING*, 8(2):215–260, April 1997.

[28] Catarci T, Chang SK, Costabile MF, Levialdi S, and Santucci G. A graph based framework for multiparadigmatic visual access to database. *IEEE TRANSACTIONS ON KNOWLEDGE AND DATA ENGINEERING*, 8(3):455–475, June 1996.

[29] Merz U and King R. Direct - a query facility for multiple databases. *ACM TRANSACTIONS ON INFORMATION SYSTEMS*, 12(4):339–359, Oct 1994.

[30] Li WS and Shim J. Facilitating complex web queries through visual user interfaces and query relaxation. *COMPUTER NETWORKS AND ISDN SYSTEMS*, 30(1-7):149–159, 1998.

[31] Clement T. Yu and Weiyi Meng. *Principles of Database Query Processing.* Morgan Kaufmann Publishers Inc., 1998.