Data/Knowledge Packets as a means of Supporting Semantic Heterogeneity in Multidatabase Systems

Doyle Weishar and Larry Kerschberg
dweishar@gmuvax2.gmu.edu, kersch@gmuvax2.gmu.edu

Department of Information and Software Systems Engineering
School of Information Technology and Engineering
George Mason University, Fairfax, VA 22030

Abstract: *Semantic heterogeneity in heterogeneous autonomous databases poses problems in instance matches, units conversion (value interpretation), contextual and structural mismatches, etc. In this work we examine some of the research issues in semantic heterogeneity and propose a novel architecture for resolving such problems. The approach involves the use of Artificial Intelligence tools and techniques to construct "domain models," that is data and knowledge representations of the constituent databases and an overall domain model of the semantic interactions among the databases. These domain models are represented as Knowledge Sources (KSs) in a blackboard architecture. This architecture lends itself to an opportunistic approach to query processing and goal-directed problem solving. We introduce the notion of Data/Knowledge Packets as a means of supporting both operational and structural semantic heterogeneity.*

1. Introduction

In this paper we deal with the problem of semantic heterogeneity in accessing heterogeneous autonomous databases. This topic has been discussed in a special issue of the *Data Engineering Bulletin* (June 1990, Vol.13, No.2) and, most recently, in the Kyoto Workshop on Interoperability in Multidatabase Systems. By semantic heterogeneity we mean that the meanings, or semantics, of objects, attributes and relationships represented in the component databases are inconsistent with respect to one another, even though they refer to the same "real-world" objects. Examples include problems in instance matches, units conversion (value interpretation), contextual and structural mismatches, etc.

We believe an intelligent query processing capability is essential to supporting semantic heterogeneity. In previous work [WeKe89, 91] we proposed a novel architecture which affords this capability. The approach involves the use of Artificial Intelligence tools and techniques to construct "domain models," that is data and knowledge representations of the constituent databases and an overall domain model of the semantic interactions among the databases. These domain models are represented as Knowledge Sources (KSs) in a blackboard architecture. This architecture lends itself to an opportunistic approach to query processing and goal-directed problem solving.

We also introduced the notion of "Data/Knowledge Packets" as a means of supporting semantic heterogeneity. This paper will extend this notion. We will briefly outline the InHead Architecture, illustrate how data/knowledge packets are used in the architecture, describe how domain models are constructed, and through an example, show how data/knowledge packets can be used as a means of supporting semantic heterogeneity in multidatabase systems.

2. The Intelligent Heterogeneous Autonomous Database Architecture (InHead)

The InHead approach is to extend the state of the art in heterogeneous DBMS interface technology by integrating Artificial Intelligence (AI) problem-solving techniques with advanced semantic data modeling techniques. The approach draws upon the flexible and opportunistic problem-solving capability of blackboard architectures, and the expressive power of the Knowledge/Data Model (KDM) [PoKe88] which allows both knowledge and data to be represented in a unified data/knowledge representation.

The InHead system incorporates an object-oriented Knowledge/Data Model, the KDM, and knowledge sources (KSs) possessing global and local domain expertise. These KSs work together to provide users with *simultaneous, multiple* viewpoints of the system at varying levels of abstraction. One KS can be looking at a user's problem from a global perspective, while another can be viewing it in terms of a local database. In this way users can more fully access system-wide data relationships.

A novel feature of InHead is a *global thesaurus* KS. The InHead active and intelligent global thesaurus provides the strategic problem-solving knowledge required to control semantic heterogeneity. The thesaurus addresses semantic heterogeneity issues in which data items may be similarly named, are related, are a subclass of, and/or are a superclass of data items located elsewhere within the system's databases. In this regard, the thesaurus plays the traditional role of data dictionary. What separates this approach from others is that the thesaurus takes on a new role: it ***actively*** works with users to reformulate queries based on knowledge associated with terms and their usage in the local databases. Figure 1 illustrates the architecture.

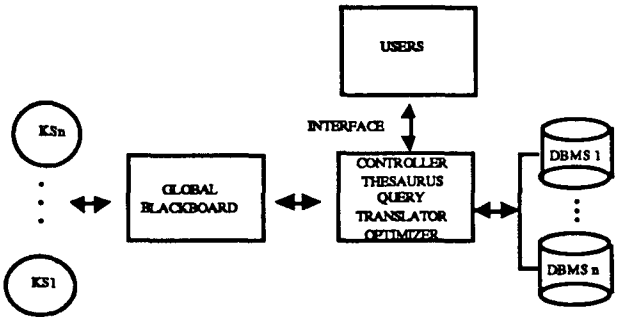

Figure 1: The Intelligent Heterogeneous Autonomous Database (InHead) Architecture

Query processing in InHead occurs along multiple viewpoints consonant with differing levels of abstraction. Therefore, it may be necessary for local database sites to understand not only the KDM specifications of the objects, but also the reasoning goals of the query processor, that is, the task for which the objects are required. Thus we contend that simple export schemata capture only a portion of the semantics we envision. Therefore we introduce the notion of "Export Data/Knowledge/Task" Schemata in which both the structural and operational semantics of the local database are expressed. The latter are culled from applications (knowledge-based and conventional) and user queries.

3. Object Encapsulation -- Data/Knowledge Packets

Local database terms, concepts, relationships, constraints and operations that have semantic variants can be "encapsulated" into an abstract object type with sufficient domain knowledge to be able to translate and interpret the appropriate meaning for an object. Data/knowledge packets are the means to encapsulate this knowledge. We suggest encapsulating the behavioral semantics of an object with its structural specification; we can then export not only object structural semantics but also object operational semantics expressed in the form of integrity constraints, methods, rules and task-specific knowledge.

Data/knowledge packets are suitable for specifying abstract object types that at the global level provide a unified viewpoint, both structurally and operationally, of semantically heterogeneous objects. Thus the local databases would remain the same, but the data/knowledge packets together with the local domain model could be used to perform semantic reconciliation at the local level first, before routing the query to the global knowledge source. This action can be seen in a situation in which there are three aircraft databases, each having the same "capacity" attribute, but all understanding capacity in different ways. A local domain KS and a data/knowledge packet, specifying the local meaning of capacity, could be used to make the initial determination of query satisfaction.

Within InHead, data/knowledge packets are formed using triples of the form <Operation,Object,Meaning> where: Operations are functions or procedures reflecting database operations, application-based operations or derived operations. Derived operations are analogous to derived data. They occur through the interactions of the various KSs on the blackboard. For example, a "Select" operation on the blackboard might result in the activation of an application-based derivation which would warrant a request for translation. That request for translation would be considered to be derived.

Objects can be database entities, attributes or operations. Allowing objects to be used to represent operations provides the capability to describe the operational semantics of the system. Meaning is set valued. If the set is non-nil, meanings can be considered as constraints, an aspect which will be illustrated in a later example.

Data/knowledge packets can be formed using single <O,O,M> triples, and as conjunctions or disjunctions. Within a data/knowledge packet triple, a component may be nil. Nil valued components are considered as "don't cares" to the multidatabase system. KSs can use data/knowledge packets stand-alone or concatenated to other system requests (e.g., to denote the meaning of a field in a "Select" operation).

Despite the similarities between the specification of a data/knowledge packet and the specification of a method in an object-oriented model, the behavior is quite different. A method performs an operation on an object to (usually) return a value. The function of a data/knowledge packet is to provide an indication of the semantics associated with the abstract object type it portrays. A data/knowledge packet behavior can be passive, by allowing itself to be inspected, or active by seeking out knowledge from the other constituents.

4. Deriving Knowledge Sources through Domain Modeling

As stated earlier, we believe that one technique to capture the knowledge necessary to form KSs and data/knowledge packets is to construct, for each database, an object-oriented *domain model* that characterizes the data, meta-data, constraints, and knowledge associated with the object types and the object (database) instances they refer to. At the local level these domain models (KSs) form the essence of the export/data/knowledge schemata. In addition, we propose constructing a global domain model that provides an understanding of the relationships among the objects in the KSs. Our approach is to construct an active and intelligent thesaurus that provides the semantic relationships among the objects in the local KSs.

To construct a domain model of a heterogeneous multidatabase system, one must study both the internals of the constituent databases and the interactions of the constituent databases. Domain analysis of constituent databases can be performed by studying the database structures (concentrating entity types and relationship types), analyzing the applications using the database, reviewing past database queries and categorizing user views of data. Domain analysis of the interactions of the constituent databases occurs in much the same fashion but on a higher level of abstraction. For a more detailed exposition of the domain modeling methodology see [GoKF89].

We note that a problem faced in federated autonomous databases is that the kind of detailed knowledge necessary to perform proper domain analyses is unavailable. Since much of what we advocate is a reengineering approach, this could also be a problem with this approach. However, because our reengineering effort is directed towards capturing semantics into modular KSs and data/knowledge packets, this problem will be mitigated to a large extent. A simple example of this would be where the domain analysis for one database was complete (in the framework described above), but the domain analysis for another yielded only the database structure (in KDM). Using the knowledge from the complete domain analysis, the reasoning mechanism could analyze both database structures and draw conclusions on the semantics of the latter database based upon its structural similarities with the former.

5 An Example -- The Artillery Movement Problem

Finally, we turn our attention to an example that illustrates how data/knowledge packets, in the framework of the InHead architecture, support semantic heterogeneity in multidatabase systems. This example will show support for object structural semantics, by identifying the different meanings of similarly named objects, and for operational semantics by disambiguating the behavior of specified abstract object types.

For this example, suppose that we have an expert system whose task is to provision 10 M110 Howitzer Weapon Systems for departure to the Middle East in 5 days. This expert system interacts with three primary databases: 1) a characteristics database which describes the physical characteristics of the component parts of weapons systems; 2) a weapons systems database which describes the components of weapons systems; and 3) a logistics database which describes the logistics support required to sustain weapons systems in combat. Secondary, but related databases are: a personnel database for crew requisitioning; a ships database for obtaining space on seagoing vessels; an Army installation database, with attributes such as name, type, and location; and an Army units database that describes a unit's location, its weapons systems, its readiness status, and its deployment status.

The expert system, which plays the role of the user in this example, has a task-oriented functional view of the problem as follows. Potentially semantic ambiguous terms are denoted in bold-face.

Overall Goal: Provision 10 M110 Howitzer Weapon Systems for departure to the Middle East in 5 days

Subgoals

1.0 Determine Availability of 10 M110 Howitzer Weapon Systems.
- 1.1 Determine the locations of such items, subject to constraint of being within 500 **miles** of Norfolk, Virginia.
- 1.2 Send requests for items to locations to hold for shipment.

2.0 Determine Availability of Logistics Support Units
- 2.1 Specialize camouflage to **desert** conditions.
- 2.2 Specialize radar to **desert** night vision.
- 2.3 Specialize rations to **high water content** rations.
- 2.4 Specialize clothing to **lightweight, chemically resistant.**

3.0 Determine Availability of Sealift Capability along the Eastern Seaboard.
- 3.1 Calculate total **weight** and **volume** for each system.
- 3.2 Provision **crews** for each system.
- 3.3 Assign **crews** and weapons to ships.

We now discuss several of the possible cases in which semantic heterogeneity is manifested in this system.

The first ambiguity occurs in the meaning of the word **miles**. The expert system may be assuming *nautical* miles while the logistics database might be assuming *statute* miles. When the logistics database sends its answers to the expert system, the data includes measurement units in its data/knowledge packets. The thesaurus is consulted for any unit translations. If ambiguity persists, the user can be consulted to provide appropriate definitions.

By focusing on Subgoal 1.0, we now look at the cooperative problem-solving aspect of InHead, which is key to detecting and disambiguating terms. After the expert system retrieves all of the M110 locations, it begins to determine if these installations satisfy the 500 mile constraint. These locations have been returned by installation name and therefore, must be converted to grid coordinates to compute their distance in statute miles from Norfolk, VA. Thus the expert system places a data/knowledge packet on the blackboard in the form of <SELECT,WPN_SYS.LOCN="FT BRAGG, NC",GR ID>.

Because the blackboard allows KSs to see and understand system goals and subgoals, they can actively contribute to the process. In this case the installation database KS, understanding locations in longitude and latitude, helps by placing <nil,INST.LOCN="FT BRAGG, NC",39°N35°W> on the blackboard. The thesaurus KS knowing that there are several instances of LOCN invokes a method to translate long/lat to grid coordinates and places <nil,LOCN="FT BRAGG, NC",12344321> on the blackboard, which is then used by the expert system to compute the distance from Ft, Bragg, NC to Norfolk, VA.

Further, the expert system might specify the task, "Provide logistic support for ten M110 howitzer systems with desert camouflage." But the logistics support database has an attribute for camouflage in terms of color combinations, rather than the term **desert**. The global thesaurus *knows* that desert colors are grey and brown, so that the semantic heterogeneity is handled easily. Also, if that information were not in the thesaurus, the expert system would engage a dialog with the user to define the term for the thesaurus.

Another instance of ambiguity involves the use of the term **crews**. Crews can be either operational crews or maintenance crews. In provisioning crews for each system, the system must know if one or both is needed. That type of information can be placed in the thesaurus. A default rule could be that operational crews take precedence over maintenance crews with the assumption that one maintenance crew can maintain several systems.

Much of the above illustrates support for structural semantics. We now continue with the example to highlight support for operational semantics.

Suppose now that the expert system's task is to compute the cost of moving three howitzer units by air, sea and rail to specified locations. To accomplish this task, the expert system works with "movements management" databases at the local sites. These databases are organized for local support, and as such, are tailored to local needs. Movements management database 1 is structured to support travel by air. The "move" operation in database 2 supports rail computations while the database 3 move operation bases its computation on ground transportation data. Data/knowledge packets allow this heterogeneity to be identified and handled. The expert system could append the data/knowledge packet <move,howitzer,{air,rail,sea}> to the "Select" operation (identifying the affected howitzer units) that it would place upon the blackboard. Database 3 could use this information to compute the cost based upon ground transportation. However, because ground transportation was not identified in the data/knowledge packet as an acceptable meaning, database 3 must send its cost data (via the blackboard) to a conversion station (e.g, a multi-way method residing in the global thesaurus) before sending the final answer back to the expert system. This communication across the blackboard thus disguises the heterogeneity of the three "move" operations.

6. Conclusions

We feel that the InHead architecture provides some unique insights and features to overcome the problems associated with semantic heterogeneity in multiple heterogeneous autonomous databases.

The construction of a domain model allows the system's constituent databases to evolve over time, yet maintain system-wide semantic consistency. The use of the "Data/Knowledge Packet" allows the encapsulation of both structural and operational semantics in data/knowledge packets. And by using the blackboard paradigm, semantic ambiguity is reduced through incremental cooperative problem-solving techniques that support multiple viewpoints of database objects simultaneously.

7. References

[GoFK89] Gomaa H, R Fairley and L Kerschberg, "Towards an Evolutionary Domain Life Cycle Model", *Proc. Workshop on Domain Modeling for Software Engineering*, OOPSLA'89, New Orleans, October 3, 1989.

[KaMK91] Kaufman, K. A. , R. S. Michalski and L. Kerschberg, "An Architecture for Knowledge Discovery from Facts: Integrating Database Knowledge Base and Machine Learning in INLEN," in *Knowledge Discovery in Databases*, (eds. G. Piatetsky-Shapiro and W. J. Frawley), The MIT Press, Cambridge, MA, 1991.

[PoKe88] Potter, W.D. and L. Kerschberg, "The Knowledge/Data Model: An Integrated Approach to Modeling Knowledge and Data," in *Data and Knowledge (DS-2)*, (R.A. Meersman and A.C. Sernadas, editors), North Holland, 1988.

[WeKe89] D. Weishar and L. Kerschberg, "An Intelligent Interface for Query specification to Heterogeneous Database," NSF Workshop on Heterogeneous Database Systems, Northwestern University, Evanston IL, December 1989.

[WeKe91] D. Weishar and L. Kerschberg, "An Intelligent Heterogeneous Autonomous Database Architecture for Semantic Heterogeneity Support," *IEEE International Workshop on Interoperability in Multidatabase Systems*, Kyoto, Japan, April, 1991.