Implemented Applications of the Carnot Heterogeneous Database System

Michael N. Huhns*
Department of Electrical and Computer Engineering
University of South Carolina
Columbia, SC 29208, USA
huhns@sc.edu

15 November 1996

Abstract

The Carnot project integrated a variety of techniques to achieve interoperation in heterogeneous environments. We describe four major applications of this project, concerning (a) accessing a legacy scientific database, (b) automating a workflow involving legacy systems, (c) cleaning data, and (d) retrieving semantically appropriate information from structured databases in response to text queries. These applications support scientific decision support, business process management, data integrity enhancement, and analytical decision support, respectively. They demonstrate Carnot's capabilities for (a) heterogeneous query processing, (b) relaxed transaction and workflow management, (c) knowledge discovery, and (d) heterogeneous resource model integration.

1 Introduction

Even as database technology has made significant inroads into real applications, nontrivial problems in information automation still remain. All too often, enterprise information systems consist of a diverse mix of applications, files, and databases that are each individually essential, but do not cohere well as a whole. Many of these systems were not designed as such, but have just evolved to keep up with new needs and technologies. This has resulted in a mix of operational systems that collectively manage huge amounts of data. The data is critical to the enterprise, but also redundant and inconsistent.

The need to access diverse information systems in a logically coherent manner translates into three technical challenges: (1) interoperability, despite heterogeneity with respect to communication and database connection protocols, query languages, logical schema access, and application semantics; (2) distribution of resources; and (3) autonomy of resources in terms of metadata and schemas, legacy applications, and closed transactions.

The Carnot architecture provides a flexible framework for addressing the challenges. It presupposes an open, standards-based, distributed computational approach based on mediated access to passive and active resources, such as databases, knowledge sources, and applications. It includes a facility for specifying constraints among resources, enabling transparent access to data at the conceptual level and strategies to maintain or restore consistency in the face of various contingencies.

The above situation is well-recognized [4, 17]. Like the Carnot project, a number of research projects have addressed this problem by developing toolkits that enable interoperation to varying extents. These are excellently reviewed and tabulated in [12], so we shall concentrate on the diverse applications of the Carnot project.

2 Overview of the Carnot Architecture

Carnot is composed of the following five major layers of services, as shown in Figure 1: semantic, distribution, support, communication, and access. The Carnot execution environment is a software component called the *Extensible Services Switch (ESS)*. The ESS is a distributed interpreter that provides access to communication, information, and application resources at a site [16]. The ESS is constructed in Rosette, an actor language enhanced with object-oriented mechanisms [1, 6]. Rosette has facilities for remote evaluation and concurrent execution, making it an effective infrastructure for coordinating information resources and transactions.

The ESS operates typically at each host as a single process that contains actors for each computation in which it is engaged. The ESSs at different hosts communicate with each other and, based on their interactions, can invoke remote operations either through specialized actors within the ESS, or by spawning separate operating system processes. Specifically, the ESS can invoke operations at multiple databases concurrently, thereby managing the gathering and combining of results.

The *Distributed Semantic Query and Transaction Manager (DSQTM)*, which physically resides inside

*The Carnot Project team included Philip E. Cannata, Nigel Jacobs, Tomasz Ksiezyk, Kayliang Ong, Wei-Min Shen, Munindar P. Singh, Christine Tomlinson, and Darrell Woelk.

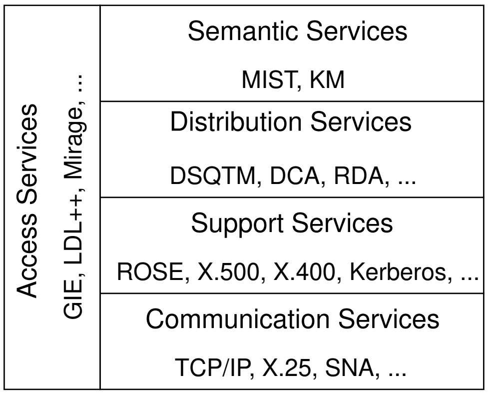

Figure 1: The Basic Carnot Layers

an ESS process, uses its data dictionaries to produce scripts that specify the distribution of queries to other ESSs or databases, and the collection and processing of results, e.g., to perform joins or domain value translations. The DSQTM includes actors that embody the protocols for accessing various resources, including Oracle, Sybase, Ingres, Objectivity, and Verity (Topic). It also has an implementation of the Relational Data Access (RDA) standard.

The distribution layer also includes the *Distributed Communicating Agent (DCA)* facility. DCA is a tool that supports the modular construction and interoperation of agents—both human and knowledge-based. Each knowledge-based agent, called a RAD agent, can perform forward and backward reasoning, and includes a frame system with multiple inheritance, distributed truth-maintenance [7], and contradiction resolution. The ESS manages communications among the agents: actors in the ESS serve as communication aides, one for each agent, and forward messages through the ESS treespace.

Lastly, distribution services include the *Declarative Resource Constraint Base (DRCB)*. The DRCB is an extended data catalog that captures interresource dependencies, consistency requirements, contingency strategies, and organizational rules. The interresource dependencies are expressed as mappings in a dictionary.

The knowledge-based RAD agents are used to capture the consistency constraints to help maintain the coherence of applications executing across autonomous information resources. The agents use models of each other and of the resources local to them so as to communicate and cooperate effectively. Resource models may be the schemas of databases, frame systems of knowledge bases, or process models of business operations. These enable relaxed, distributed transactions to execute concurrently across heterogeneous databases that previously had incompatible semantics. Thus the appearance and effect of homogeneity among heterogeneous resources is obtained.

The semantic services consist of a suite of tools for enterprise modeling, model integration, data cleaning, and knowledge discovery. The *Model Integration and Semantics Tool (MIST)* is used to generate mappings and consistency constraints, which form the basis for semantic mediation among information resources in the distribution services. MIST relies on a *common ontology* [8], which represents the concepts and their relationships characterizing a domain of interest. Database schemas, even from the same application domain, implicitly involve distinct concepts, which makes it difficult to relate them. However, by relating different database schemas to a common ontology, we can semantically relate the schemas with each other, and thereby enable interoperation of the underlying databases. MIST can work with a common ontology expressed in Cyc [10], or in Carnot's own knowledge representation tools called KRBL. Carnot was one of the pioneers of an ontology-based approach to interoperation.

The semantic services also include the *Knowledge Representation Base Language (KRBL)* as a tool for representing and accessing ontologies. This tool has a simple frame-based representation of such knowledge in the form of n-ary relations, metaclasses, and metarelations.

The *Knowledge Miner (KM)* is a semantic tool used for knowledge discovery. It includes symbolic inductive learning and statistical clustering techniques, which it combines with the LDL++ deductive database environment (described below). The knowledge discovery methods infer patterns and regularities from information resources and check consistency between information and corresponding models [13]. The discovery is guided by expectations about the nature of the information and its embedding in the application.

The access services provide mechanisms for personal and group interaction with Carnot services, including user interface software and the *Logical Data Language (LDL++)*, which provides a Prolog-like rule-based language optimized for database access [11]. Figure 2 shows some example configurations with the major components and their relationships.

3 Legacy System Access

This application involves accessing data from a legacy database for scientific decision support at Eastman Chemical Company (ECC). ECC maintains information about chemical research, development, manufacturing, marketing, and customer contacts in several large, incompatible databases.

Queries to these databases have been very difficult, requiring an expert to assist in retrieving information and taking days to weeks to satisfy a single query. This delay proves particularly expensive to research chemists, who must repeat experiments if the information produced in previous experiments cannot be found. Redoing an experiment is not only time-consuming, but also can cost several thousand dollars.

We briefly describe the domain in order to better motivate our approach. The domain consists of

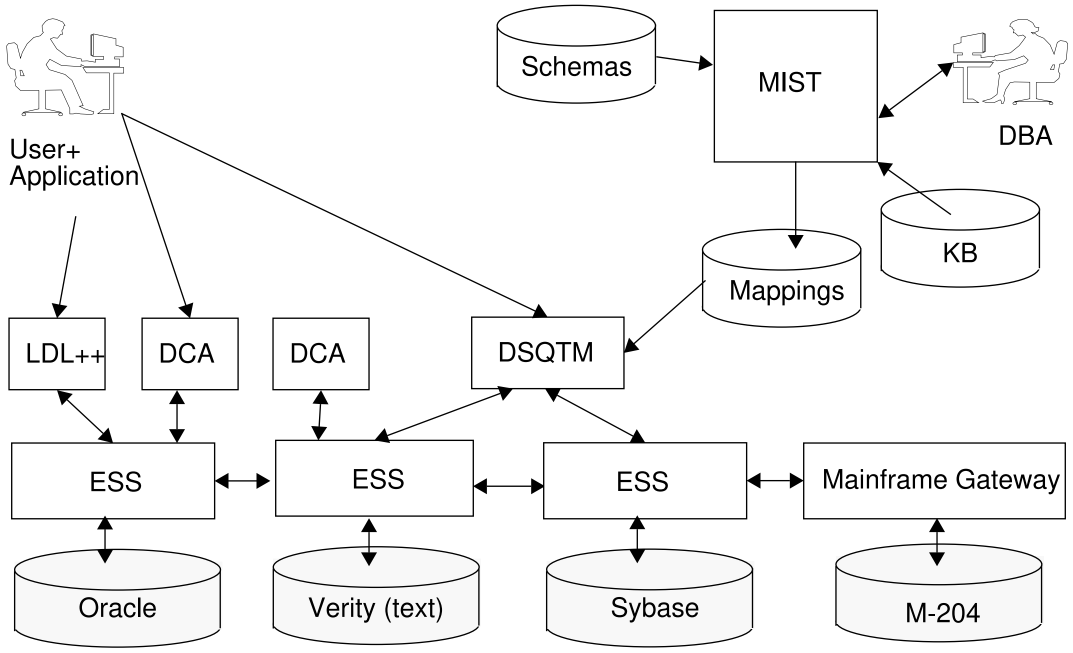

Figure 2: Schematic Carnot Configuration

chemical compounds, which are identified by unique names and defined as compositions of other chemicals. Roughly about 100 different chemical and physical tests are performed on different compounds to measure their properties of interest to various applications (the details of these tests are proprietary and, in any case, not of interest to our readers). The `composition` table represents the main entity; there is a table for each experiment recording (few to several) values in different columns. Since the number of chemicals in a compound is not limited, the ECC database designers used a flattening representation in which each compound may be represented through a set of tuples in the `composition` table, each tuple carrying the identifier of the compound and the given chemical and its amount. The above database is primarily of interest to scientists; another database contains information of interest to marketers, and uses a different key, but we shall not discuss in any detail that here.

Several of the Carnot components were used to implement the solution shown in Figure 3. The system is in operation and being extended. When it is set up, a knowledge base is created containing representations of all the logical views of interest. In addition, dictionaries are created that relate the logical views to the physical tables and columns in a polymer research database (PDRS).

Our implementation includes two user interfaces, one that is forms-based and uses LDL++, and the other natural-language based and provided by MCC's Knowledge-Based Natural Language project (KBNL) [2, 3]. LDL++ supports the formulation of complex queries as logical rules. The LDL++ compiler gathers the rules for each query, generates compact SQL statements, and dispatches them to the database server via the ESS. LDL++ represents `composition` and the experimental result tables as predicates, as for example

```
composition(Id: int, Code: string, quantity: float)
```

One interface is a form by which scientists can find compounds that satisfy some range conditions on the chemicals that compose them. For example, one might ask for all compounds that contain 5-10% of `A` and 50-67% of `B`, yielding the following constraints:

```
[compositionC(Id,'A',range(5.0,10.0)),
 compositionC(Id,'B',range(50.0,67.0))]
```

From the input range values and domain knowledge, the LDL++ application validates the constraints and possibly augments them. If successfully validated, the augmented constraints are converted to SQL and executed on the database.

```
SELECT DISTINCT TB_COMPOSIT_1.PEL_ID
FROM TB_COMPOSIT TB_COMPOSIT_1, TB_COMPOSIT TB_COMPOSIT_2
WHERE TB_COMPOSIT_1.PEL_ID  = TB_COMPOSIT.PEL_ID
      AND TB_COMPOSIT_1.AMO_CODE =  'A'
      AND TB_COMPOSIT_1.AMOUNT >= 5
      AND TB_COMPOSIT_1.AMOUNT <= 10
      AND TB_COMPOSIT_2.AMO_CODE =  'B'
      AND TB_COMPOSIT_2.AMOUNT >= 50
      AND TB_COMPOSIT_2.AMOUNT <= 67
```

The KBNL system takes English inputs and, after appropriate interactions with the user, produces a high-level SQL query, and hands it over

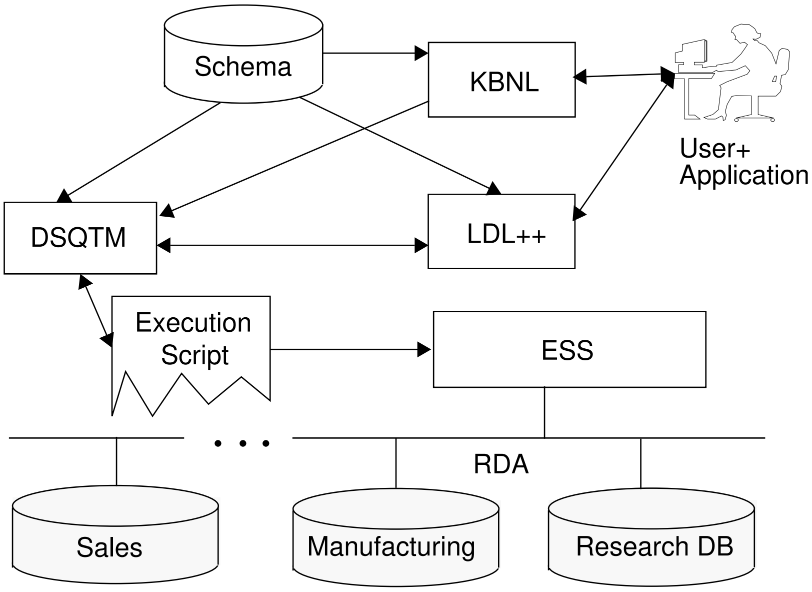

Figure 3: Architecture for Legacy System Access

to Carnot for processing. Example English inputs include (i) `Find polymers with elongation of 3.3 and creep of 4.4`, (ii) `What is the creep for polymer A?` and so on. KBNL interacts with the user to disambiguate their information request to the level of the conceptual schema, e.g., to determine that the user cares about `chemical resistance break elongation` and `adhesive creep at time 50`, making the query effectively be:

```
Find polymers with
     chemical resistance break elongation of 3.3
     and adhesive creep at time 50 of 4.4
```

KBNL then produces an SQL query, involving logical tables and columns, and forwards it to Carnot:

```
SELECT *
FROM POLYMERS
WHERE POLYMERS.PEL_ID = CHEMICAL_RESISTANCE.PEL_ID
AND POLYMERS.PEL_ID = ADHESIVE_CREEP.PEL_ID
AND CHEMICAL_RESISTANCE.ELONGATION_AT_BREAK = 3.3
AND ADHESIVE_CREEP.TIME_050 = 4.4
```

The above SQL query is received and processed by the DSQTM. The DSQTM generates and the executes the following low-level SQL query:

```
SELECT DISTINCT TB_COMPOSIT.PEL_ID
FROM TB_COMPOSIT , TB_CHEM_RES , TB_ADH_CREP
WHERE TB_CHEM_RES.BRK_ELNG  =  3.3
      AND TB_COMPOSIT.PEL_ID  =  TB_CHEM_RES.PEL_ID
      AND TB_COMPOSIT.PEL_ID  =  TB_ADH_CREP.PEL_ID
      AND TB_ADH_CREP.TIME_050 = 4.4
```

In order to produce the above query, the DSQTM uses knowledge about KBNL's logical view, and its mapping to the physical view. The interesting part of this knowledge is in the form of articulation axioms. In this application, most of the axioms give a straight one-to-one mapping. However, some axioms encode that the objects of the logical view are represented as split across multiple rows in the physical table–these are the *id convention* axioms below. These axioms are used in a queries similar to the one we showed above using LDL++, so we shall not discuss them again.

```
# Mapping KBNL and PDRS views to common (ECC) view
KBNL ADHESIVE_CREEP TIME_050 <==>
     ECC_ADHESIVE_CREEP TIME_050
KBNL ADHESIVE_CREEP TIME_150 <==>
     ECC_ADHESIVE_CREEP TIME_150
KBNL CHEMICAL_RESISTANCE ELONGATION_AT_BREAK <==>
     ECC_CHEMICAL_RESISTANCE ELONGATION_AT_BREAK

KBNL ADHESIVE_CREEP POLYMERS <==>
     ECC ECC_ID_CONVENTION ECC_NAME
KBNL CHEMICAL_RESISTANCE POLYMERS <==>
     ECC ECC_ID_CONVENTION ECC_NAME
PDRS TB_ADH_CREP PEL_ID <==>
     ECC ECC_ID_CONVENTION PEL_NAME
PDRS TB_CHEM_RES PEL_ID <==>
     ECC ECC_ID_CONVENTION PEL_NAME


# ECC_NAME is represented as multiple columns
  in TB_COMPOSIT
PDRS TB_COMPOSIT PEL_ID  <==>
     ECC_ID_CONVENTION ECC_NAME
PDRS TB_COMPOSIT AMP_CODE <==>
     ECC_ID_CONVENTION ECC_NAME
```

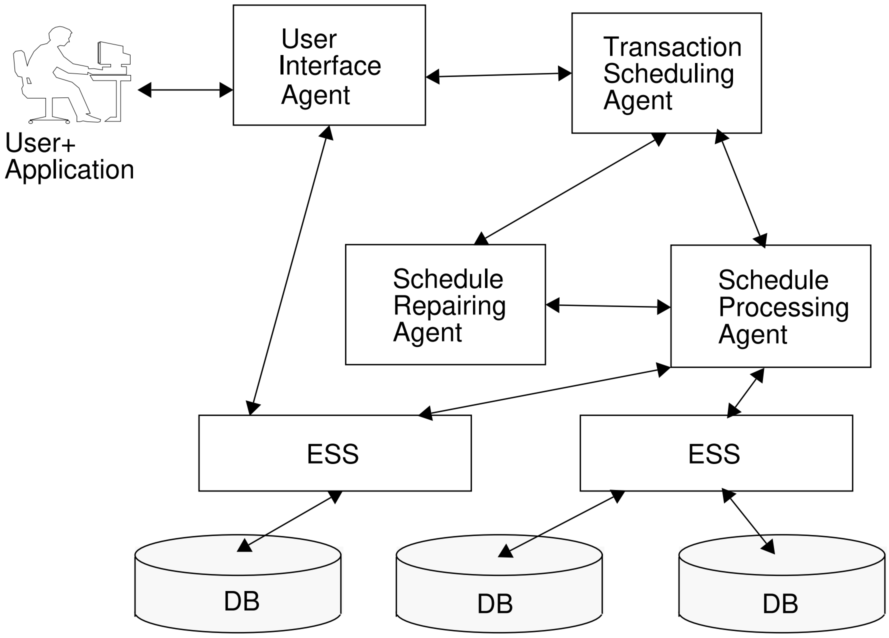

Figure 4: Architecture for Workflow Automation

```
PDRS TB_COMPOSIT AMOUNT   <==>
     ECC_ID_CONVENTION ECC_NAME
PDRS TB_COMPOSIT AMO_CODE <==>
     ECC_ID_CONVENTION ECC_NAME

# TB_COMPOSIT joins with every table that has
  a PEL_NAME column
PDRS TB_COMPOSIT PEL_ID  <==>
     ECC_ID_CONVENTION PEL_NAME
```

Our implementation at ECC involved a commercial implementation of the RDA protocol, which enables access to backend nonrelational and older relational databases.

For the above architectural framework to be effective, the logical view of the database taken by the interfaces must agree with the logical view supported by the DSQTM. Since there are more than 100 tables, many involving concepts that were arcane to us, we formulated a shared representation of the database schemas. We defined a number of scripts to map these shared schemas into the knowledge base used by the natural-language interface to understand and disambiguate English queries, by LDL++ to form predicate descriptions, and by the DSQTM to build its dictionaries.

4 Workflow Automation

Workflows, especially database-oriented workflows, have emerged as the leading paradigm for structuring complex computations in heterogeneous information environments [5]. Carnot was one of the pioneers in workflow management from a database perspective (as opposed to the more traditional organizational or groupware perspective). Workflows are important wherever there are complex, long-running, flexible, interactive flows of information and control.

Carnot was applied to the DS-1 *service provisioning* activities of Ameritech, a telecommunications company. Service provisioning refers to the task of connecting a customer to the system—assigning a connection to them, making sure the physical infrastructure exists, and updating the various databases. This task could take up to two weeks, involving tens of operations on over a dozen operational (legacy) systems. Our aim was to prototype a system through which the throughput and delay could be improved.

Our system consists of four DCA agents—a user interface agent and three RAD knowledge-based agents. The user agent assists the user in ensuring that the service request is valid. When it completes, it sends a message to the scheduling agent. The scheduling agent determines a workflow schedule—initially, this is the normal case of the execution of the activity. The schedule processing agent executes the tasks in this schedule by invoking operations on the backend systems concurrently. Some of these operations require significant protocol conversion, e.g., in generating messages that can be sent via custom interfaces to legacy systems or mainframes. Exception conditions are captured declaratively in the schedule repair agent. These conditions determine when the composite activity should be aborted and when different component transactions should be retried or compensated. Additional details of this application are available in [15].

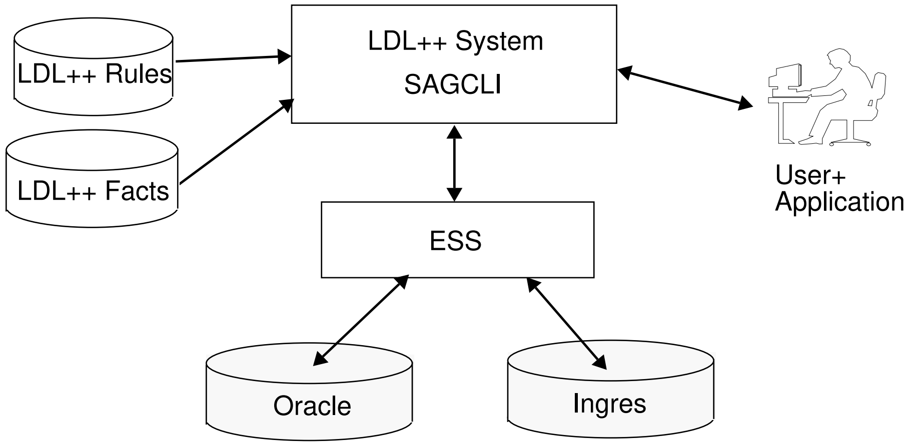

Figure 5: Architecture for Data Purification

The success of this application derives from its taking into account an entire run through the system, not just focusing on some of the pieces. The expert system technology that we used here was key to rapid prototyping, but is by no means essential. However, capturing execution conditions is a challenge for conventional programming, and is eased by a rule-based language that includes support for maintaining dependencies among decisions.

5 Data Purification

Data is considered a vital business asset and the quality of data plays a critical role in the quality and efficiency of business operations. Bellcore began a data quality project in partnership with the Carnot project. Poor data quality (1) impedes workflow automation because of more frequent and unnecessary exceptions during processing, and (2) makes it harder to provide good customer service due to the inconsistency and incompleteness of data [14]. To perform data validation and cleaning, we determined that three capabilities are required:

- *Database Access.* Telephone companies have a wide range of heterogeneous databases, which must be accessed before any data can be validated and cleaned.
- *Specification of Complex Validation Rules and Queries.* Data is validated based on the rules that define whether the given data is correct or is suspected to be incorrect.
- *Rapid Refinement of Validation Specification.* The system might take several iterations to verify correctness and cleanliness of data with respect to the specifications.

LDL++ and ESS were identified as the core Carnot technologies to fulfill the above requirements. As shown in Figure 5, the ESS provides access to the Oracle database. LDL++'s declarative nature facilitates specification of the validation conditions as a set of LDL++ rules. A graphical user interface allows users to dispatch different validation queries, review the results, and take corrective actions. Just as for ECC, LDL++ generates SQL, which is executed via the ESS. Databases can be validated for different types of constraints, including

- *Domain Value Constraints*—column values must be from a certain range.

```
channel(length,range(0.0,10.0))
channel(conductivity,range(50.0,117.0))
```

- *Quantitative Constraints*—values in different columns must satisfy certain numerical constraints, such as the two below. Violations of the first constraint—if a channel connects to a piece of equipment, then the channel must terminate at that equipment—are obviously errors. Conversely, data that satisfies the second constraint—if a loop links to a cable, then the start location of the loop equals the end location of the cable—is also likely to be erroneous.

```
link(channel,equipment) =>
     equal(endLoc(channel), loc(equipment))
link(loop,cable) =>
     equal(startLoc(loop), endLoc(cable))
```

- *Uniqueness Constraints*—some column values must be unique, e.g., a cable only links to one loop.

```
link(loop1,cable), link(loop2,cable) =>
     equal(loop1,loop2)
```

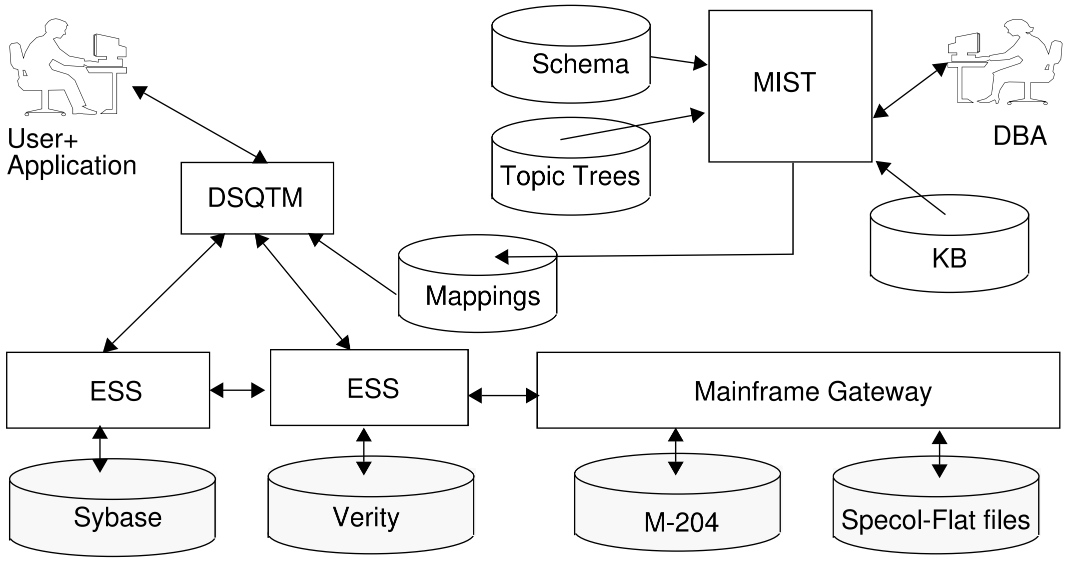

Figure 6: Architecture for Combined Structured Data and Text Access

- *Referential Integrity Constraints*—the existence of a value in one place may presuppose its existence elsewhere in the system, possibly in another database.

```
link(loop,cable) => loopInfo(loop, \_, \_)
```

6 Combining Structured Data and Text

The U.S. Department of Defense (DoD) has several text as well as traditional structured databases that must be used in concert. The *Verity Topic* system is used for text retrieval. A Topic query takes the form of a weighted tree of concepts, including target words and phrases. Verity processes the query against text databases and presents a ranked list of matching documents.

The structured databases are often maintained by different individuals, and employ different designs, database software, and platforms. Ideally, users should corroborate their findings from documents with information from these databases. But this requires them to learn about different schemas, to master multiple database query languages, and to be able to interpret the results.

Carnot was applied to this case as shown in Figure 6. At compile-time, textual and relational data sources were integrated via a common conceptual model [9]. At run-time, database queries were managed by a distributed network of database agents [16].

A simplified Topic tree capturing the concept of a MiG29 airplane is shown in the top left of Figure 7. Intuitively, an article might be about MiG29s if it mentions enough of the terms associated with MiG29 in the tree. The top right of Figure 7 shows a simple ontology with concepts pertaining to weapons, fighter aircraft, and the human designers of weapons. The bottom right shows two database tables with information about fighter aircraft and persons. There might be an entry for MiG29 in the first table, and for its designer, Mikoyan, in the second table.

The Topic tree and database tables are related through the ontology. First, a Topic concept tree parser is used to create internal representations of Topic trees that can be used by MIST. Second, MIST is then used to map each concept in the trees to corresponding concepts in a common ontology, providing a semantic interpretation for each tree, and identifying concepts from different trees having the same meaning. Relational database schemas are also mapped to the common ontology, yielding executable linkages among the Topic trees and structured databases.

When a user issues a Topic query, the DSQTM uses the above mappings to produce a corresponding set of SQL queries. The original query is executed on the text engine, the SQL queries are executed on the structured databases, and the DSQTM fuses the results. One effective fusion method was to produce a hypertext document from the articles returned by the text search. Selected words, e.g., “MiG29,” are given hypertext links to results of queries from the structured databases.

7 Conclusions

The above applications include some of the major and common business needs that heterogeneous database systems must address: (1) accessing data from a high-level view, coordinating transactions across systems, data cleaning, and fusing traditional data with nonstandard data. Carnot addressed these different problems through a uniform framework. The problem of accessing data through high-level views was well-known before Carnot, although few commer-

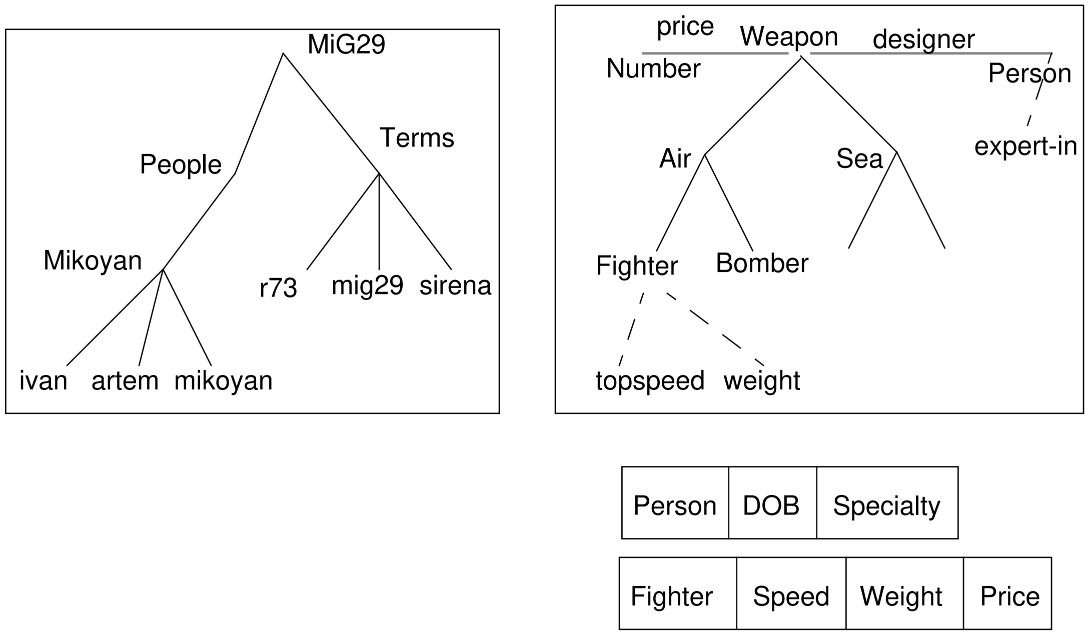

Figure 7: Relating Topic Trees and Database Schemas

cial solutions are available even today. The growth of the workflow and data cleaning industries over the past few years is phenomenal, but these problems were just barely being understood when Carnot implemented the corresponding applications. Nontraditional, unstructured data such as text is common in many practical applications of computing, yet, a few years ago, unstructured data was not given its due importance within the database community.

Carnot contributed a number of interesting ideas to database research, including

- Development of tools to perform resource integration, even among resources of different models.
- Use of intelligent agent technology to coordinate transactions and workflows, in particular encapsulating exception conditions for workflows.
- Use of actor technology with scripting languages to coordinate distributed activities.
- Use of deductive database technology to provide natural access to data, especially for data intensive applications such as knowledge discovery and data purification.

Carnot addressed some of the most challenging problems in making heterogeneous information systems function effectively. It developed new theories, prototyped them, and deployed them. However, despite the "success" of the applications, they are still prototypes, and more effort is required before they can be considered commercial quality.

References

[1] Gul A. Agha. *Actors*. MIT Press, Cambridge, MA, 1986.

[2] Jim Barnett, Kevin Knight, Inderjeet Mani, and Elaine Rich. Knowledge and natural language processing. *Communications of the ACM*, 33(8):50–71, August 1990.

[3] Jim Barnett, Juan Carlos Martinez, and Elaine Rich. A functional interface to a knowledge base: An NLP perspective. Technical Report ACT-NL-393-91, Microelectronics and Computer Technology Corporation, Austin, TX, 1991.

[4] Omran A. Bukhres and Ahmed K. Elmagarmid, editors. *Object-Oriented Multidatabase Systems: A Solution for Advanced Applications*. Prentice Hall, 1996.

[5] Dimitrios Georgakopoulos, Mark Hornick, and Amit Sheth. An overview of workflow management: From process modeling to workflow automation infrastructure. *Distributed and Parallel Databases*, 3(2):119–152, April 1995.

[6] C. Hewitt, P. Bishop, and R. Steiger. A universal modular actor formalism for Artificial Intelligence. In *IJCAI*, 1973.

[7] Michael Huhns and David M. Bridgeland. Multiagent truth maintenance. *IEEE Transactions on Systems, Man, and Cybernetics*, 21(6):1437–1445, 1991.

[8] Michael N. Huhns, Christine Collet, and Wei-Min Shen. Resource integration using a large knowledge base in Carnot. *IEEE Computer*, 24(12):55–62, December 1991.

[9] Michael N. Huhns, Nigel Jacobs, Tomasz Ksiezyk, Wei-Min Shen, Munindar P. Singh, and Philip E. Cannata. Integrating enterprise information models in Carnot. In *International Conference on Intelligent and Cooperative Information Systems (CoopIS)*, May 1993.

[10] Douglas Lenat and R.V. Guha. *Building Large Knowledge-Based Systems: Representation and Inference in the Cyc project*. Addison Wesley, Reading, MA, 1990.

[11] Shamim Naqvi and Shalom Tsur. *A Logical Language for Data and Knowledge Bases*. W.H. Freeman Publishers, New York, NY, 1989.

[12] Evaggelia Pitoura, Omran A. Bukhres, and Ahmed K. Elmagarmid. Object-oriented multidatabase systems: An overview. In *[4]*, chapter 10. 1996.

[13] Wei-Min Shen, Bharat Mitbander, KayLiang Ong, and Carlo Zaniolo. Using metaqueries to integrate inductive learning and deductive database technology. In *Proceedings of the AAAI Workshop on Knowledge Discovery from Databases*, August 1994.

[14] Amit Sheth, Christopher Wood, and Vipul Kashyap. Q-Data: Using deductive database technology to improve data quality. In Raghu Ramakrishnan, editor, *Applications of Deductive Databases*. Kluwer Publishers, 1994.

[15] Munindar P. Singh and Michael N. Huhns. Automating workflows for service provisioning: Integrating AI and database technologies. *IEEE Expert*, 9(5), October 1994. Special issue on *The Best of CAIA'94* with selected papers from Proceedings of the 10th IEEE Conference on Artificial Intelligence for Applications, March 1994.

[16] Christine Tomlinson, Philip E. Cannata, Greg Meredith, and Darrell Woelk. The extensible services switch in Carnot. *IEEE Parallel and Distributed Technology*, pages 16–20, May 1993.

[17] Gio Wiederhold. Mediators in the architecture of future information systems. *IEEE Computer*, 25(3):38–49, March 1992.