Conceptual Models and Architectures for Advanced Information Systems [1]

Larry Kerschberg and Doyle J. Weishar

kersch@gmu.edu.
Center for Information Systems Integration and Evolution
Department of Information and Software Engineering
MSN4A4, George Mason University, 4400University Drive,
Fairfax, VA, 22030-4444

dweishar@globalinfotek.com
Global InfoTek, Inc
156 East Maple Avenue
Vienna, VA 22180

Abstract. This paper addresses several issues related to the use of conceptual modeling to support service-oriented, advanced information systems. It shows how conceptual modeling of information resources can be used to integrate information obtained from multiple data sources, including both internal and external data. The notion of an *intelligent thesaurus* is presented and a meta-model of the thesaurus is developed. It is then used to create a three-layer architecture consisting of the actual data source schemas, a "wrapped" object-oriented abstraction of the schemas expressed in terms of the thesaurus primitives, and an integrated version which serves as the federation schema. The sharing of information among constituents is also addressed, and a special export schema - the export data/knowledge/task schema - is proposed that ties together the objects, their constraints, and their usage rules. Knowledge sharing among constituents during cooperative query processing is accomplished using *data/knowledge packets*.

Key Words: Intelligent Thesaurus, Knowledge Sharing, Ontology

1 Introduction

We live in a world that is becoming increasing distributed and service-oriented, and this is reflected in the information systems we design, build, manage and use. We rely on these systems to support our everyday activities through electronic mail, document interchange and other collaboration technologies. The Internet and World Wide Web (WWW) [1] allow us to access multimedia information located throughout the world. Clearly, these new technologies provide enormous opportunities for posting, finding, organizing and sharing vast amounts of data, information and knowledge.

Electronic business is becoming the premiere Web application, and presents a vital complement to the normal business activities conducted by corporations and individuals. For some companies, such as Amazon.com and E-Trade.com, the *virtual marketspace* is their *only* site. New paradigms are emerging for

[1] L. Kerschberg and D. Weishar, "Conceptual Models and Architectures for Advanced Information Systems," *Applied Intelligence*, vol. 13, pp. 149-164, 2000.

e-business, both business-to-business and customer-to-business, allowing consumers to be linked directly with providers, and thereby bypassing the traditional "middleman." This *disintermediation* of the relationship between customers and producers is welcome, but it presents problems for users to find the appropriate suppliers for a product of interest. Thus, *infomediation* is required to allow consumers to locate and contact potential suppliers.

In the real world, our service-oriented approach to doing business leads us to delegate both *responsibility* and *authority* for certain negotiations and decisions to our representatives or agents, such as real-estate agents, stock brokers, secretaries, etc. Correspondingly, we can have agents that that act on our behalf in the electronic marketspace.

The major issues confronting consumers of on-line information and services include access and availability of information resources and services, confidence in the veracity of the information provided, and an assessment of the trustworthiness and reliability of the provider. Thus, our collection of trusted and reliable agents should be able to:

1. Search for, acquire, analyze, integrate and curate data from multiple heterogeneous distributed sources [2-8],

2. Inform when new data of special interest becomes available [9],

3. Negotiate for, purchase and receive information, goods and services [10-14],

4. Explain the relevance, quality and reliability of that information [15, 16], and

5. Adapt, learn and evolve in response to changing conditions [17-25].

What is needed is an architecture that incorporates the concepts of infomediation with the flexibility of agents. In this paper, we present an architecture to create *advanced information systems* that offers an info-mediated, agent-facilitated and service-orient approach to the access, collection, integration, configuration and management of heterogeneous information sources.

This paper focuses on the conceptual modeling of data, information and knowledge, for the purpose of integrating both internal and external data, into an information architecture supported by an *intelligent thesaurus*. Section 2 of the paper presents a framework for advanced information systems with a focus on active services to support information mediation. Section 3 discusses the role of an intelligent thesaurus in infomediation, and uses the Knowledge/Data Model [26] to specify the thesaurus meta-model. The modeling of local schemas in terms of object-oriented concepts is presented, as well as the integration and sharing of this information among a federation of sites. Section 4 presents our conclusions.

2 A Framework for Advanced Information Services

During the mid- to late-1980's, Kerschberg was involved in promoting the integration of research in Artificial Intelligence, Logic Programming, Information Retrieval and Database Systems, calling it *Expert Database Systems* [27-30]. Since then, substantial progress has been made in making Database Management Systems more active and knowledgeable, while at the same time facilitating access to large databases by Expert Systems and other knowledge-based systems.

More recently, research has focused on the Intelligent Integration of Information (I*3) [2, 31, 32]. Here the problem is to *access* diverse data residing in multiple, autonomous, heterogeneous information sources, and to *integrate,* or *fuse,* that data into coherent information that can be used by decision makers. To make the problem even more challenging:

1. data may by *multimedia* (video, images, text, and sound);

2. sources may store the data in diverse formats (flat files, network, relational-, or object-oriented databases);

3. the meaning of data, i.e., *data semantics* may conflict across multiple sources;

4. the data may have differing temporal and spatial granularities;

5. much of the *interesting* and *valuable* data may reside outside the enterprise, in the open-source literature accessible via subscription services, broadcast services, and on the World Wide Web (WWW) [1]; and

6. the data may be of uncertain quality, and the reliability of the source may be questionable.

It is becoming increasingly apparent that one cannot expect to solve I*3 and other large-scale system problems with a *monolithic* solution. Rather, the solution should consist of smaller components, each having the requisite knowledge to perform its tasks within the larger problem-solving framework.

2.1 Three Layer Infomediation Architecture

Bowman [33, 34] describes a three-layer architecture for scalable Internet resource discovery, proposed by the Internet Research Task Group. Similarly, the DARPA-sponsored research program on the "Intelligent Integration of Information" (I*3) produced a three-layer Reference Architecture consisting of various types of mediation services, including facilitation and brokerage services, mediation and integration services, and wrapping and data access services.

Table 1 denotes a three-layer infomediation architecture which provides access to heterogeneous data sources, including those on the WWW, together with some of the important I*3 services supporting each layer [12].

Table 1: Active Infomediation Services

Information Layer	Layer Service	Active Services
Information Interface Layer	Users perceive the available information at this layer and may query and browse the data. This layer must support scalable organizing, browsing and search.	Thesaurus Services Yellow Pages
Information Management Layer	Responsible for the replication, distribution, and caching of information. Responsible for the integration of information gathered from multiple sources	Mediation Services Active View Services Integration Services
Information Gathering Layer	Responsible for the collecting and correlating the information from many incomplete, inconsistent, and heterogeneous repositories.	Federation Wrapping Services Data Quality and Inconsistency Management

Figure 1 depicts our approach to the three-layer architecture for the application domain of logistics; the architecture is general enough to apply to most enterprises. The information architecture incorporates the three information layers of Table 1 consisting of:

1) *Information interface layer* where users access the system, formulate queries, collaborate in problem-solving activities, initiate pull scenarios and receive information from push scenarios. Users have access to their local databases and work through local views. We assume that collaboration mechanisms and tools exist at this layer;

2) *Information management layer* where objects, mediated active views, and information in an Intelligent Thesaurus are integrated, managed, replicated, updated. This layer mediates between the information interface layer and the information gathering layer, allowing users to perceive an *integrated information space*, when in reality, data resides in multiple heterogeneous databases and information sources. A mediated view of data is provided at this layer and user views are materialized from the mediated view.

   The Real-Time Information Processing and Filtering process constantly monitors the system for events of importance to enterprise activities, and informs users, the mediated view, and the Intelligent Thesaurus should these events occur.

   The Intelligent Thesaurus contains meta-data and knowledge associated with enterprise resources. The thesaurus provides the conceptual model of the objects represented and materialized from multimedia heterogeneous and distributed sources. There is an ongoing process to identify objects, their inter-relationships, and associated knowledge. The lineage of an object is also important for traceability and data quality concerns.

3) *Information gathering layer* where data from diverse, heterogeneous inter-networked information sources are accessed.

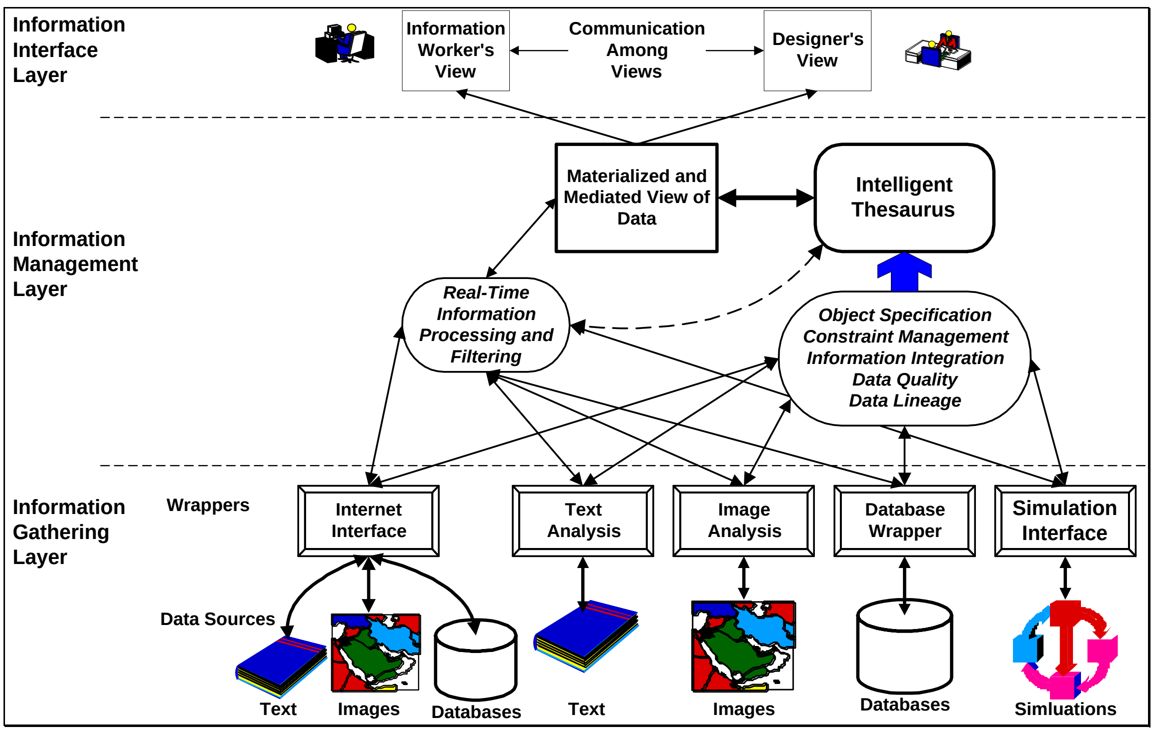

Figure 1: Layered Information Architecture & Processes

2.2 Thesaurus Services

The *intelligent thesaurus* [35-37] is an *active* data/knowledge dictionary capable of supporting multiple ontologies and allowing users to formulate and reformulate requests for objects. The intelligent thesaurus is similar to the thesaurus found in a library; it assists users in identifying similar, broader or narrower terms related to a particular search term, thereby increasing the likelihood of obtaining the desired information from the information sources. In addition, active rules and heuristics may be associated with object types as well as their attributes and functions.

2.3 Mediation Services

Mediation refers to a broad class of services associated with I*3 [2, 31, 38, 39]. One example is the mediation of temporal data of differing granularity. This is of particular importance in the context of multidimensional databases and data warehousing applications, where historical data is integrated and analyzed for patterns and interesting properties.

A *temporal mediator* [40-42] consists of three components: 1) a repository of *windowing functions* and *conversion functions*, 2) a time unit thesaurus, and 3) a query interpreter. There are two types of windowing functions: the first associates time points to sets of object instances, and the other associates object instances to sets of time points. A conversion function transforms information in terms of one time unit into that in terms of some other time unit. The time unit thesaurus stores the knowledge about time units (e.g., names of time units and relationships among them). The time-unit thesaurus stores concepts such as the seasons, fiscal year definitions, and calendars, and effects translation of these time units into others.

Users pose queries using the windowing functions and desired time units using a temporal relational algebra. To answer such a user query, the query interpreter first employs the windowing functions together with the time unit thesaurus to access the temporal data from the underlying databases and then uses the time unit thesaurus to select suitable conversion functions which convert the responses to the desired time units. Thus, a temporal mediator provides a simple, yet powerful, interface that supports multiple temporal representations in a federated environment. Temporal mediators may also be used to compare historical databases such as those needed for auditing and data warehousing purposes.

2.4 Active View Services

Active views are motivated by the need to mediate between users and the plethora of enterprise data and information being generated. Active views can be used to define complex objects, events and conditions of interest. Thus, active views mitigate the need for users to constantly issue queries to verify the existence of certain events or condition in the enterprise, or to be bombarded constantly with irrelevant information. The active view automates this task by compiling user object specifications into rules monitored by active databases [43-46], or polling queries issued against traditional enterprise database systems. These rules can also be given to agents, called rovers, that comb the WWW seeking relevant information. Active views also provide active caching [9, 47, 48] of materialized views so as to support subscription services, notification services, and to update user profiles. Further, materialized views can be updated automatically.

2.5 Federation Services

Our approach to federations of systems consists of an I*3 federated distributed client/server architecture [49] in which the constituent systems maintain authority and autonomy over their data, while at the same time sharing certain information with the federation. Client software and/or server software is provided to members so that they can interface existing information systems with the federation. Each constituent system has a Federation Interface Manager (FIM) which provides certain services such as federal to local translation services, and wrapping services. There is a Federation Manager (FM) that provides federation services such as an *integrated conceptual view* of data, as well as temporal mediation and spatial mediation services. This was proposed in an Independent Architecture Study for the Earth Observing System Data and Information System (EOSDIS) [35, 50, 51], and NASA is now fostering a form of federation for EOSDIS.

2.6 Data Quality and Inconsistency Management

In any environment of multiple information resources one would expect that sources would overlap in providing similar but inconsistent data. Inconsistencies are detected during the process of integration, and *harmonization agents* are engaged to resolve them. The concept of the harmonization agent is incorporated in the Multiplex [16, 52] proof-of-concept system which considers the reliability and quality of the conflicting information sources, and resolves conflicts in a way that increases the overall value of the information [15, 53-55].

3 Thesaurus Services for Infomediation in the InHead Architecture

In this section we present a conceptual model for thesaurus services to support the infomediation process to assist users in finding information located in multiple heterogeneous databases. The goal is to provide an *intelligent thesaurus* capable of providing information about objects, their relationships, constraints, contexts, as well as knowledge (rules and heuristics) governing those objects.

The thesaurus, therefore, provides a *conceptual model* of the objects in the information space, as well as how the objects may be used, that is, in which contexts and by whom. This implies that our conceptual model should capture the underlying concepts, their relationships to other concepts, their properties, constraints associated with the concepts, and the knowledge associated with the role and use of a concept in the model. The thesaurus can assist in resolving semantic heterogeneity issues encountered when creating a integrated view of concepts spanning multiple databases.

In order to model the thesaurus concepts, we use the Knowledge/Data Model (KDM) [26] that incorporates an object-oriented view of data, together with knowledge regarding its usage. The next section introduces the conceptual model for the thesaurus, and its representation in the KDM. The concepts presented form the core of the InHead Architecture developed by Weishar in his doctoral dissertation [37], and described in [36, 56]. InHead refers to the Intelligent Heterogeneous Autonomous Database Architecture and prototype system that incorporates the intelligent thesaurus, and it is used to support the three-layer infomediation approach discussed earlier.

A prototype of InHead has been implemented using Macintosh Common Lisp, version 2.0, as a base language, and the Blackboard Technology Group's "GBB", which provides a generic blackboard application programming environment. InHead makes effective use of the GBB facilities including: object management, blackboard database administration facilities, built-in control paradigms, the ability to import knowledge sources written in Common Lisp, CLOS, and OPS5, and graphics facilities. The federation for the prototype includes three databases: (1) an Army aircraft database; (2) an Air Force aircraft database; and (3) a Navy aircraft database. The major objectives of the proof-of-concept prototype were to illustrate: (1) the utility of intelligent thesaurus in resolving semantic heterogeneity in a multidatabase system; (2) the effectiveness of InHead objects in expressing the semantics of federation constituents; (3) the feasibility of data/knowledge packets for cooperative query processing; and (4) the capability of InHead to successively reformulate queries by using the Intelligent Thesaurus.

3.1 Thesaurus Conceptual Model

The thesaurus consists of terms, relationships among terms, and their constraints, and is modeled after a library thesaurus. In addition, we provide knowledge associated with the terms.

Broader and narrower terms are depicted by object generalization and specialization hierarchies, respectively, and are normally formed using "is–a" relationships. Broader terms denote an expanded scope of the object–type's meaning. For example, an object–type "airplane" could have the broader term "aircraft." A broader term subsumes the object-type. Formally, a broader term B is a proper superset of a given object–type A (i.e., $A \subset B \Leftrightarrow$ for all $x[x \in A \Rightarrow x \in B]$ where $A \neq B$). Conversely, narrower terms restrict the scope of meaning. A narrower term "jet" could be specified for "airplane". Again, formally, a narrower term N is a proper subset of a given object–type A (i.e., $N \subset A \Leftrightarrow$ for all $x[x \in N \Rightarrow x \in A]$ where $N \neq A$).

Similar terms are depicted using "is–similar–to" relationships. The concept of similarity is not as formal as other concepts. Similar terms "resemble" or "approximate" the properties of the given object type, i.e., they are alike but not identical. The degree of similarity is left up to the interpretation of the system designer when constructing the object–type, and to the user when analyzing query results. For example, a similar term for an object–type "cannon" is "howitzer". Both share the properties of propelling projectiles through the air, but do so by differing means.

Homonyms are treated as special cases of similar terms. Determining, for example, whether a car is an automobile or the first element of a Lisp list requires the attachment of context–sensitive heuristics to the object–type.

Synonyms and Preferred terms (aliases) are denoted via the "is–eq–to" relationships. Two terms are synonymous if all of their attributes that are values are equal and all of their attributes that are objects are recursively value–equal. (Note that this definition is a less restrictive version of object value equality. Object identifiers do not have to be the same to establish synonymy.) Formally, a term S is equal to a given term A (i.e., $A = B \Leftrightarrow$ for all $x[x \in A \Rightarrow x \in B]$). **Antonyms**, which are opposites of synonyms, are depicted using "is–op–of" relationships.

Individual instances of the above categories can be shown in ***Classification Hierarchies***, which are formed by applying "is–instance–of" relationships to class objects. Similarly, object–types grouped together to form higher–order object–types are organized into ***Aggregation Hierarchies***, which are formed by applying "is–part–of" relationships to the group of class objects. Object–types that are part of a group of objects each sharing common interests but neither changing individual attributes nor forming higher–order objects (for example, showing student object–types as members of the computer–club object–type) are depicted using ***Membership Hierarchies***. Membership hierarchies are formed by applying "is–member–of" relationships to the class–objects.

As mentioned above, object–types have ***Object Roles*** within a federation. These are enumerated as values of the "known_as" attribute in thesaurus object–types (the specification of which follows later in this section).

Knowledge of constituent term ***constraints*** is included in object ***methods***. Constraint knowledge is limited herein to application-specific integrity constraints, versus data model specific and schema specific database integrity constraints. Methods also specify any heuristics associated with the constituent term. Heuristics can also be used to specify object relationships that supersede or modify object relationships. For example, the heuristic "A Merchant Marine ship is a Navy ship in wartime," specifies the "is–a" and "is–eq–to" relationships. **Temporal Knowledge** in the thesaurus is also specified as a constraint.

The thesaurus also maintains ***Object Views*** which specify logical partitioning of the instances of Federation Interface Manager (FIM) objects, either for security reasons or for query processing. Also within the InHead thesaurus are specifications for global, FIM, local, and external federation object views. Each view provides a slightly different conceptualization of the thesaurus term and is used by both users (in a browse mode) and by federation components to discover the semantics of federation terms.

There are two consequences of the above compilation of knowledge. First, the object specification process yields knowledge of federation replicated, fragmented and/or overlapping data/knowledge. Second, the process provides a type of schema integration service.

While the InHead architecture does not call for a global schema, the combination of the federation–level and FIM–level knowledge sources in effect becomes *an active and intelligent global schema*. The InHead approach provides federation constituents with autonomy, hides the underlying system heterogeneity from users, and permits database administrators to specify the relationships among concepts by means of the knowledge-rich thesaurus.

3.2 The Knowledge/Data Model

The Knowledge/Data Model [26] is an extension of the semantic data model and draws heavily upon the features of the functional data model , object-oriented paradigm, and knowledge-based systems. The KDM models the semantics of an enterprise, including data semantics, as captured by semantic data models and knowledge semantics, as captured in knowledge-based systems. The significant features of the KDM data model are:

- The incorporation of heuristics to model inferential relationships.

- The capability to organize these heuristics and to associate them with specific items involved in the inferential relationships.

- The capability to incorporate heuristics and constraints in a tightly coupled (unified) manner.

- The ability to define inferred (virtual) objects.

- A unified representational formalism for knowledge and data.

- A mechanism that allows for abstract knowledge typing, that is, handling rules and constraints as objects.

The following are the semantic primitives available in the KDM.

- **Generalization:** Generalization (the inverse of which is **specialization**) provides the facility in the KDM to abstract similar object-types into a more general or higher-level object-type (an object-type is a collection of related objects). This is done by means of the "is-a" relationship (e.g., the object-types "student" and "soldier" and generalized into the "person" object-type). This generalization hierarchy establishes the inheritance mechanism (e.g., "student" and "soldier" inherit the properties and methods of "person").

- **Aggregation:** Aggregation (the inverse of which is **decomposition**) is an abstraction mechanism where an object is related to the components that constitute it via the "is-part-of" relationship (e.g., the "name", "rank" and "serial_number" are part of "soldier_data").

- **Classification:** Classification (the inverse of which is **instantiation**) provides a means whereby specific object instances can be grouped together and considered to be an object-type. This is accomplished through the use of the "is-instance-of" relationship (e.g., "Blackhawk" and "Huey" and specific instances of "Army_Aircraft").

- **Membership:** Membership is an abstraction mechanism that specifically supports the "is-a-member-of" relationship between objects or object-types (e.g., "maneuver_force" is an object-type containing "infantry_battalion" and "artillery_battalion" members).

- **Constraint:** This primitive is used to place a constraint on some aspect of an object, operation, or relationship via the "is-constraint-on" relationship. Both implicit (e.g., a specified cardinality of the set-type object-type "task_force") and explicit (e.g., the "salary" attribute of object-type "soldier" is restricted to < $30,000).

- **Heuristic:** This primitive is used to attach an heuristic via the "is-heuristic-on" relationship (e.g., the next hill to be taken can be inferred from the available firepower, the enemy capability, and the weather). Heuristics are expressed in the form of rules.

- **Method:** This primitive is used to model the behavior of object-types and to manipulate object-types. For example, an object-type might invoke a "compute-pay" method in order to derive a soldier's pay in wartime.

- **Temporal:** The temporal relationship is used to model specific task or event oriented object-types that are related by synchronous or asynchronous characteristics (e.g., the planning and conduct of a battle). Synchronous objects are related to other synchronous objects by either the predecessor or successor relationship. Asynchronous objects are related to other asynchronous objects by a concurrent or parallel notion. Temporal primitives are also used for task planning and workflow analysis.

3.3 The Knowledge/Data Language

The most generic construct in the KDM is the object type and is specified in the KDL template depicted in Figure 2, which shows a general template for an object-type (class) specification employing the KDL. KDL reserved words are shown in uppercase letters. Identifiers shown in lowercase letters are place holders for user input. Optional items in the template are enclosed in square brackets, and at least one of each of the items contained in curly brackets must be part of the specification

The optional characteristics of an object-type correspond to the KDM modeling primitives while class-name uniquely identifies the object type. Any object-type may be defined as a specialized (derived) form of one or more other object-types, called supertypes. And any object-type (or class) may be viewed as an encapsulation of the following KDL functions:

- Attributes are stored functions holding information or properties of an object.
- Members are multi-valued or association functions.
- Constraints are Boolean functions.
- Heuristics are functions or rules.
- Methods are functions that are more general than heuristics in that they can take any number of parameters and allow limited side-effects (such as producing output) to occur.

For a further discussion of the KDM and KDL readers may refer to [57-59].

```
object-type ::=
    OBJECT-TYPE: class-name HAS
        [SUPERTYPES:            // Generalization
            class-name { , class-name};]
        [SUBTYPES:      // Specialization
            class-name { , class-name} [HIDING function-list];]
        [ATTRIBUTES:            // Aggregation
            {attribute-name: type-name
                [WITH CONSTRAINT: constraint ] ; }+ ]
        [MEMBERS:       // Membership (Association)
            {member-name: [SET OF | LIST OF] class-name
                [INVERSE OF member-name [class-name)]]
                [WITH CONSTRAINT: constraint ] ; }+ ]
        [CONSTRAINTS:   // Knowledge to enforce integrity
            { constraint; }+ ]
        [HEURISTICS:            // Knowledge to derive/infer information
            {heuristic}+ ]
        [METHODS:       // Specifications of computations & behavior
            { method; }+ ]
END class-name;
```

Figure 2: KDL Object Type Specification Template

3.4 KDM Specification for the Intelligent Thesaurus Metamodel

The thesaurus metamodel expressed in the KDM/KDL is depicted in Figure 3:

OBJECT–TYPE **THESAURUS** HAS
OID: {System Defined}
SUPERTYPES: {Object–Type}
SUBTYPES:
{Thesaurus_Object,
Knowledge_Source_Object}
ATTRIBUTES:
DESCRIPTION: {String}

Figure 3: The THESAURUS Meta–Object–Type Specification.

The THESAURUS meta–object–type specification delineates object–type (class) inheritance through the SUPERTYPES and SUBTYPES slots. Note that inheritance within this context includes multiple inheritance. Within the federation there could be any number of nested federations. If that were the case, more general thesauri would be listed in the SUPERTYPES slot of the THESAURUS object–type. THESAURUS object–types have two subtype object–types, THESAURUS_OBJECT object–types and KNOWLEDGE_SOURCE_OBJECT object–types, which serve to populate the Intelligent Thesaurus. In this meta–object–type specification the DESCRIPTION attribute allow the object–type to be annotated in free text, a capability which can be useful to users browsing federation concepts.

The THESAURUS_OBJECT Object–Type Specification

THESAURUS_OBJECT object–types are used to describe the knowledge of constituent terms and of their usage. Figure 4 shows the THESAURUS_OBJECT object–type meta–schema. The specification of a THESAURUS_OBJECT meta–object–type immediately follows in Figure 5.

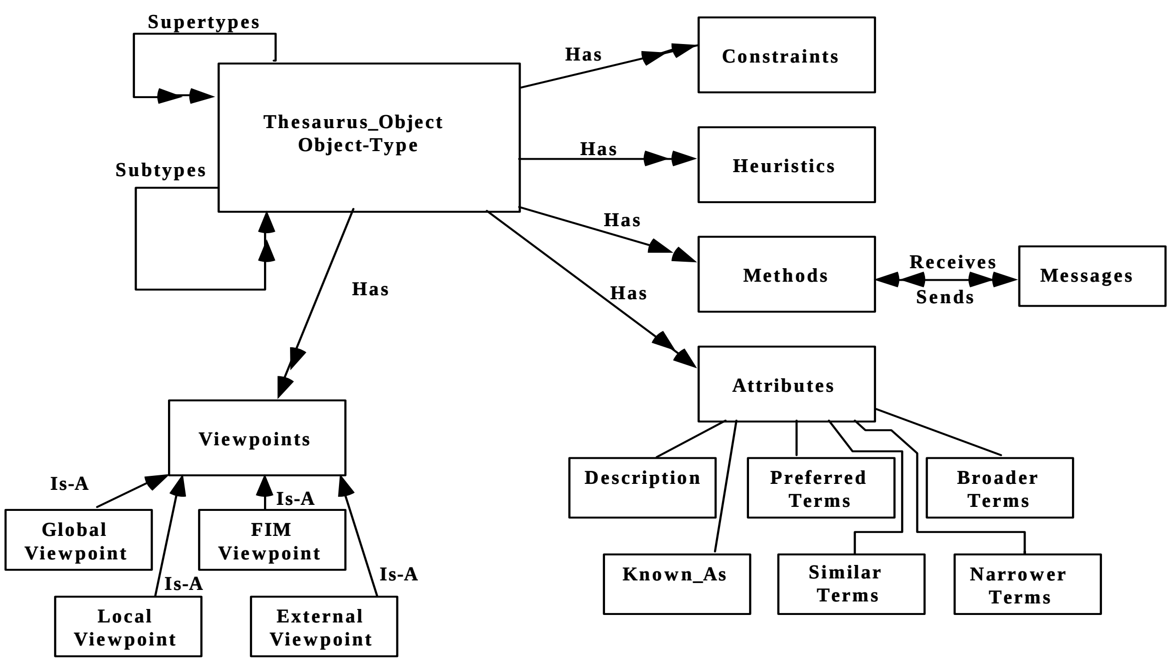

Figure 4: The THESAURUS_OBJECT Meta-Schema

OBJECT–TYPE **THESAURUS_OBJECT** HAS

OID: {System Defined}
SUPERTYPES: {Object–Type}
SUBTYPES: {Object–Type}
ATTRIBUTES:
    DESCRIPTION: {String}
    BROADER OBJECTS: {Object–Type}
    NARROWER OBJECTS: {Object–Type}
    SIMILAR OBJECTS: {Object–Type}
    PREFERRED OBJECTS: {Object–Type}
    KNOWN_AS: {SET OF Roles}
VIEWPOINTS: [[Global | FIM | Local | Ext]]
CONSTRAINTS: {SET OF Constraints}
HEURISTICS: {SET OF Heuristics}
METHODS: {SET OF Methods}

Figure 5: The THESAURUS_OBJECT Meta–Object–Type Specification.

Again, object–type (class) inheritance is specified through the SUPERTYPES and SUBTYPES slots. In many ways, the inheritance paths are duplicated by the ATTRIBUTE slots BROADER OBJECTS, NARROWER OBJECTS, SIMILAR OBJECTS, and PREFERRED OBJECTS in that these attribute values indicate subsumption (is–a), composition (part–of) relationships, equality (is–eq–to) and default values, respectively. There is an advantage to this redundancy, however, in that it provides another access path structure that goes beyond strict class inheritance associated with SUPERTYPES and SUBTYPES slots. This redundancy allows the user to focus on an object-type, and to query its properties such as narrower, broader, similar, and preferred objects directly, rather than traversing hierarchies.

This is not necessarily true when listing BROADER OBJECTS and NARROWER OBJECTS. For example, for an object class called *Navy_Fighters*, a superclass *Aircraft* could be specified. An object class *Carrier_Based_Aircraft* could be listed as a broader object since *Carrier_Based_Aircraft* would expand the scope of aircraft to include the object classes *Navy_Tankers* and *Navy_Helicopters*. Yet *Carrier_Based_Aircraft* would not qualify as a superclass because *Navy_Fighters* would include instances of both Carrier_Based and Ground_Based Navy_Fighters. *Carrier_Based_Aircraft* attributes and methods would not be inherited by Ground_Based Navy_Fighters.

The KNOWN_AS attribute indicates the “ROLE” that the object plays within the federation. Each object within the federation takes on a characteristic “ROLE”, which is a context–sensitive description of its structure. An object can have different roles depending on the level(s) at which it is represented within the federation. For example, it can be an entity or an attribute, in the database sense, at the local level. Additionally, an object can have multiple roles.

The Knowledge_Source_Object Object-Type Specification

The Knowledge_Source_Object object-types are used to encapsulate an abstraction of an entire federation constituent. It brings together information regarding the constituent with respect to the others in the federation. This object-type effectively “wraps” an entire data source, a federation constituent, into a KDM specification which serves as the source’s “local schema” object model at the federation level. Note the attribute CONS_LOCN denotes the constituent’s location, and the attribute SCHEMA refers to the set of object types “encapsulated” by the knowledge_source_object.

OBJECT–TYPE **KNOWLEDGE_SOURCE_OBJECT** HAS
OID: {System Defined}
SUPERTYPES: {Object–Type}
SUBTYPES: {Object–Type}
ATTRIBUTES:
    CONS_LOCN:{String}

SCHEMA:{SET OF Object-types}
CONSTRAINTS: {SET OF Constraints}
HEURISTICS: {SET OF Heuristics}
METHODS {SET OF Methods}

Figure 6: The KNOWLEDGE_SOURCE_OBJECT Meta-Object-Type Specification

3.5 Information Integration in the InHead Architecture

In this section we present examples showing how the KDM-based intelligent thesaurus object types can be used to create conceptual models of objects from a local database, integrate them, and present a global conceptual view of the information space.

Knowledge within InHead is organized along three levels: Federation Manager (global) level; FIM level; and at the local level (see Figure 7). At the local level the schemas are characterized in their orignal format, in this case, three relations for NAVY bombers, fighters, and utility aircraft and their respective attributes.

The FIM level supports a KDM object-oriented specification of the local schema into the NAVY Database Knowledge Source consisting of the object types *Navy_Aircraft*, with supertype *Aircraft*, and subtypes *Bombers, Fighters* and *Utility*, and each of these subtypes have their own object type specifications at the FIM Level. The specifications show as their supertype *Navy_Aircraft*, they have their local attributes and inherit global attributes from their supertype. Note the KNOWN_AS attribute has the value "Relation" denoting the role played by the object type in the local schema. The conceptual modeling processes that transform the local-level relations to FIM-level object types involve both *translation* and *abstraction*. The translation process provides mapping to-and-from local to FIM representations. The abstraction process identifies those concepts to be represented at the FIM level.

The process of incorporating FIM-level object types into the Federation Manager Level is one of *integration*. Here the FIM objects are related to the Thesaurus Knowledge Source. The Federation Manager Level knows about an object type *Aircraft*. The supertype of *Aircraft* is *Vehicle*. The subtypes of *Aircraft* are *Navy_Aircraft*, *Army_Aircraft*, and *AF_Aircraft*, with the assumption that an integrated armed services federation is being specified. Note in the specification for Aircraft the NARROWER OBJECTS attribute has an enumerated set of values: *Transport*, *Cargo*, *Fighters*, *Fxd_Wing*, and *Rotary*. The SIMILAR OBJECTS have values *Plane*, *Flying_Machine*, and *Airplane*, while the PREFERRED OBJECTS have values *Bird* and *Ship*. These terms can be used to expand or restrict a query posed in terms of the Thesaurus Knowledge Source.

3.6 Knowledge Export in the InHead Architecture

Traditionally, federated multidatabase architectures have included the notion of an export schema. An export schema denotes the objects and functions that federation constituents are willing to share with other constituents. Classical export schemas have been limited to proper subsets of a local conceptual schema, and thus, have been constrained primarily to structural descriptions.

The InHead export schema expresses both structural and behavioral semantics, including constraints and knowledge. Because query processing in InHead occurs along multiple viewpoints consistent with differing levels of abstraction, it is necessary for local database sites to understand not only the structural and behavioral semantics of objects in other databases, but also the tasks for which the requested objects are required. Since the behavioral semantics of an object-type are typically dispersed among application programs, it is necessary to associate a particular task with a specific application program. This is not possible within a traditional export schema. Therefore, we introduce the notion of an "Export Data/Knowledge/Task" Schema, EDKTS for short, in which both the structural and operational semantics of the local database are linked to specific federation tasks by means of heuristics. In this way, the recipient of the EDKTS will understand the object type specifications, the terminology expressed in terms of FIM-level concepts, and the rules, heuristics, and constraints associated with the using the objects in the EDKTS.

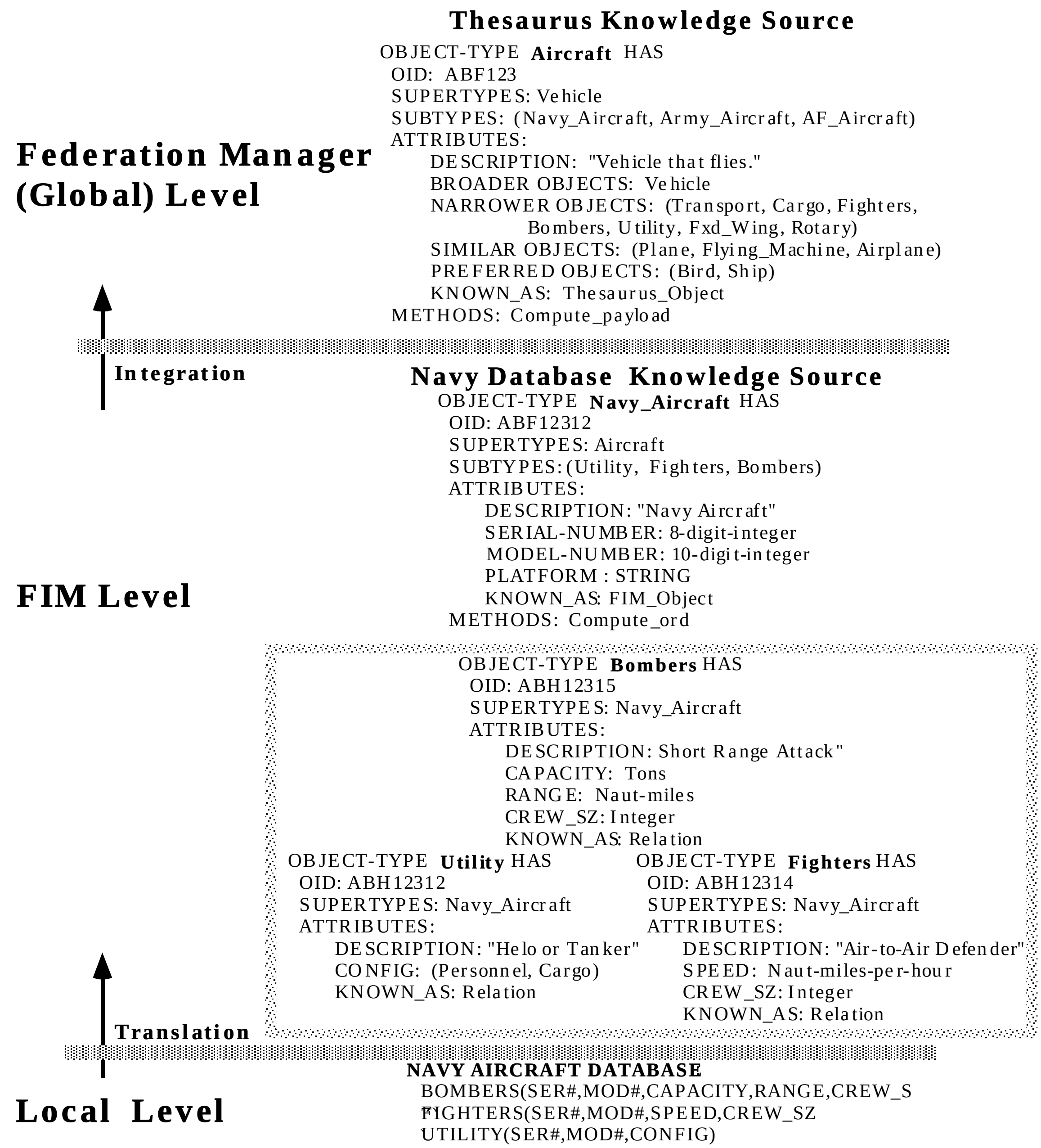

Figure 7: Three Level Specification – Local, FIM and Federation

An EDKTS is encapsulated as a specialized domain dependent knowledge source. Figure 8 depicts an example EDKTS for Navy_Aircraft that provides insight into their construction. The EDKTS for the FIM is circled at the top of figure 8.

Construction begins by modeling the local schema structures in terms object types. Here, the “Bombers” relation is represented as a *Bombers* object-type with its attributes appearing as slot names (SER# and MOD#) do not appear since they are inherited from *Navy_Aircraft*. For brevity, only *Bombers* is illustrate). The next step is to capture the behavioral semantics of the relation by representing knowledge of the relation’s local context. Local constraints and views are encapsulated as constraints in the FIM_Object. Then, local application programs that operate on the relation are listed as methods. Finally, these programs are associated with federation-wide tasks by means of heuristics. For example, heuristic H1 states that, if the federation’s goal is to “Compute aircraft readiness in wartime,” then Naval Reserve bomber assets must also be included in the local computation of bomber readiness. Thus, the “Bomber” relation’s data (i.e., its structural semantics) and knowledge (i.e., its behavioral semantics) are associated with the task at hand. Upon completing this process for all object-types involved, the local FIM designates a proper subset the object-oriented schema as its Export Data/Knowledge/Task Schema for the federation.

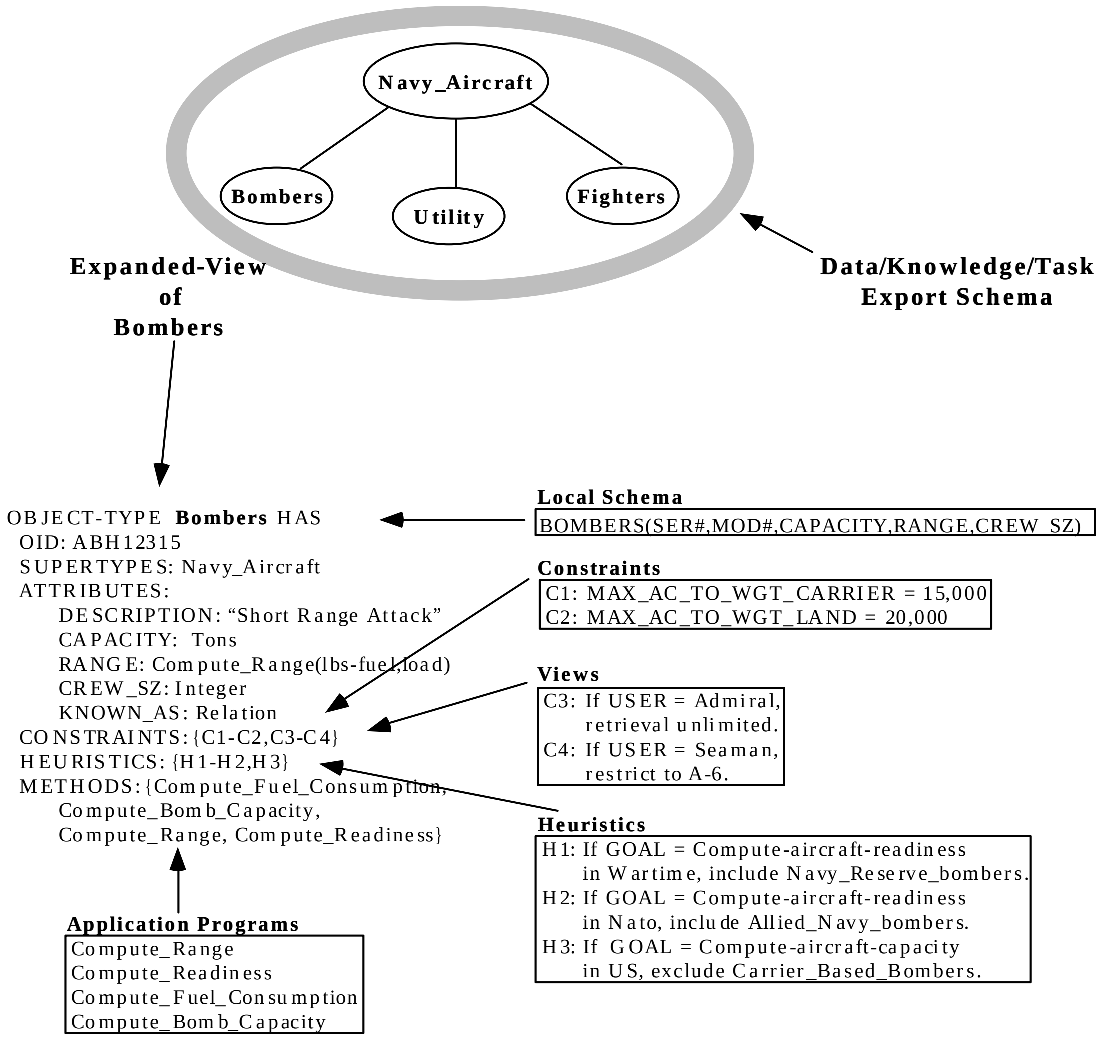

Figure 8: A Sample Export Data/Knowledge/Task Schema for a Federation Interface Manager.

3.7 Data/Knowledge Packets for Cooperative Query Processing in InHead

Users poses queries to InHead using the terminology available at the federal level, or by issuing queries to their local FIM. In the InHead prototype system, the user's query is posted on a global blackboard and individual FIMs provide partial answers to the query based on the data and knowledge they have. Federation Interface Managers, each representing a constituent, exchange information in *data/knowledge packets*, which are used to communicate partial results among FIMs and the Federation Manager.

InHead query processing is a cooperative and opportunistic process with each InHead component actively monitoring the global blackboard so that it might assist in satisfying a user query. Components assist in the process in two ways. They either establish the meaning of terms, or help other components to establish the meaning of terms. InHead components must recognize a term's semantics in a particular context. This is accomplished by comparing unknown terms with known terms whose contextual semantics are represented in various knowledge sources. If there is a match, then the component (e.g., a FIM) can inform the Federation Manager (FM), via a knowledge source activation on the blackboard, that it understands the meaning of the term. If there is a partial match or no match, the component requires additional knowledge from other federation components to help establish a term's meaning.

Within InHead, this knowledge interchange is performed using *data/knowledge packets***,** which are a means of encapsulating object structure, relationships, operations, constraints, and rules into a meaningful unit, or packet. These packets are dynamically constructed and exchanged between the Federation Manager and Federation Interface Manager components in order to establish the contextual semantics of a term. There are three uses for data/knowledge packets during query processing: 1) to "request" additional term semantics from other federation components, 2) to provide a "response" to a data/knowledge packet request, and 3) to "volunteer" additional semantics of terms to other federation components on an unsolicited basis.

Because federation components are independent, they are allowed to unilaterally initiate data/knowledge packet dialog. Thus, data/knowledge packet dialog can occur at any time during query processing. Therefore, knowledge interchange using data/knowledge packets can have a significant impact on the depth, scope and length of InHead query processing. Consequently, there must be a system-wide control mechanism that limits these knowledge interchanges. This control function is exercised on two levels.

At the federation component level, each component keeps a list of data/knowledge packets sent and received. This precludes duplication and ensures eventual termination of data/knowledge packet dialog between pairwise federation components. At the system level, the Federation Manager (FM) can oversee system-wide query processing, since federation components must interact via the blackboard. Based upon its strategic problem-solving knowledge, the FM may choose to halt all or part of InHead query processing at any time. There is also an additional safeguard at the system level. Since there is no guarantee that a federation component will receive a response to a request for data/knowledge packet, the federation component will not become available for activation again by the blackboard control mechanism if no response data/knowledge packet is provided.

4 Conclusions

This paper has presented a multi-layered federated architecture for advanced information systems. The systems access data from multiple heterogeneous and autonomous sources, integrate that data, and present a unified view of data to users. This multi-layered view allows users to search an integrated name space, to pose queries using those terms, and to receive answers in the common terminology.

Conceptual modeling of data, information and knowledge in advanced information systems is crucial to capture the diverse semantics of the concepts, and contexts in which the data is used. The terminology must be captured in an active intelligent dictionary, which we term the *intelligent thesaurus*. The intelligent thesaurus provides an integrated terminological space in which terms are treated as objects, which are organized into classes, or object types, with each class having attributes, operations, relationships, and constraints.

The InHead architecture is presented and the Knowledge/Data Model is used to model the intelligent thesaurus constructs. The process of constructing the intelligent thesaurus can be both bottom-up and top-down. In the bottom-up approach, for each local constituent database, an abstraction process is to model the objects, relationships and constraints to be shared with other federation constituents. These are modeled using the thesaurus meta-model, and an export data/knowledge/task schema is published as the federation view for that constituent. Similarly, export schemas are created for each constituent. In the top-down approach, the global terminology may be chosen based on some consensus among the constituents and the federation. Then the links to the FIM export schemas are made to create the full thesaurus.

The next step in creating the global terminological space is to integrate the individual exported views into the global view. This is accomplished by means of the thesaurus object attributes such as supertypes, subtypes, narrower_terms, broader_terms, similar_objects, methods, etc. The paper presents an example for creating the local, FIM and Federation level views.

Users can then query the system using this integrated terminology, and the system decomposes the query into subqueries to the local constituent systems. The translation of federal-level queries to local database languages is handled by the Federation Manager and Federation Interface Managers. In the InHead prototype system, the user's query is posted on a blackboard and individual systems provide partial answers to the query based on the data and knowledge they have. Federation Interface Managers, each representing a constituent, exchange information in *data/knowledge packets*, which are used to communicate partial results among FIMS and the Federation Manager.

Finally, as we design and build advanced information systems that combine multimedia data from multiple, autonomous and heterogeneous information sources, we need to use conceptual modeling tools to create a repository of terms by which users can find, retrieve, classify, store and disseminate data, information and knowledge. The intelligent thesaurus is an important tool to support these advanced architectures.

5 Acknowledgements

This research was sponsored in part by the Defense Advanced Research Project Agency (DARPA) within the Intelligent Information Integration (I*3) and Advanced Logistics Programs under contract numbers N0014-92-J-4038 and N00600-96-D-3202, respectively.

6 References

[1] T. Berners-Lee, R. Cailliau, A. Loutonen, H. F. Nielsen, and A. Secret, "The World-Wide Web," *Communications of the ACM*, vol. 37, pp. 76—82, 1994.

[2] G. Wiederhold, "Foreword to Special Issue on the Intelligent Integration of Information," *Journal of Intelligent Information Systems*, vol. 6, 2/3, pp. 93-97, 1996.

[3] Y. Arens, C. A. Knowblock, and W.-M. Shen, "Query Reformulation for Dynamic Information Integration," *Journal of Intelligent Information Systems*, vol. 6, 2/3, pp. 99-130, 1996.

[4] S. Dao and B. Perry, "Information Mediation in Cyberspace: Scalable Methods for Declarative Information Networks," *Journal of Intelligent Information Systems*, vol. 6, 2/3, pp. 131-150, 1996.

[5] R. Davis and R. Smith, "Negotiation as a Metaphor for Distributed Problem Solving," *Artificial Intelligence*, vol. 20, pp. 63-109, 1983.

[6] H. Garcia-Molina, Y. Papakonstantinou, D. Quass, A. Rajaraman, Y. Sagiv, J. Ullman, and J. Widom, "The TSIMMIS Approach to Mediation: Data Models and Languages," presented at Proc. International Workshop on Next Generation Information Technologies and Systems, Naharia, Israel, 1995.

[7] D. Kuokka and L. Harada, "Integrating Information via Matchmaking," *Journal of Intelligent Information Systems*, vol. 6, 2/3, pp. 261-279, 1996.

[8] W. W. Chu, H. Yang, K. Chiang, M. Minock, G. Chow, and C. Larson, “CoBase: A Scalable and Extensible Cooperative Information System,” *Journal of Intelligent Information Systems*, vol. 6, 2/3, pp. 223-259, 1996.

[9] L. Seligman and L. Kerschberg, “A Mediator for Approximate Consistency: Supporting "Good Enough" Materialized Views,” *Journal of Intelligent Information Systems,* vol. 8, pp. 203 - 225, 1997.

[10] L. Kerschberg, “Knowledge Rovers: Cooperative Intelligent Agent Support for Enterprise Information Architectures,” in *Cooperative Information Agents*, vol. 1202, *Lecture Notes in Artificial Intelligence,* P. Kandzia and M. Klusch, Eds. Berlin: Springer-Verlag, 1997, pp. 79-100.

[11] L. Kerschberg and S. Banerjee, “The DPSC Electronic Marketspace: The Impact of the Internet and the Web on Electronic Commerce and Logistics,” Center for Information Systems Integration and Evolution, George Mason University, Fairfax, Technical Report February 1997 1997.

[12] L. Kerschberg, “The Role of Intelligent Agents in Advanced Information Systems,” in *Advances in Databases*, vol. 1271, *Lecture Notes in Computer Science,* C. Small, P. Douglas, R. Johnson, P. King, and N. Martin, Eds. London: Springer-Verlag, 1997, pp. 1-22.

[13] L. Kerschberg and S. Banerjee, “An Agency-based Framework for Electronic Business,” in *Cooperative Information Agents III*, vol. 1652, *Lecture Notes in Artificial Intelligence*, M. Klusch, O. Shehory, and G. Weiss, Eds. Berlin, et al.: Springer-Verlag, 1999, pp. 254-279.

[14] A. Brodsky, L. Kerschberg, and S. Varas, “Resource Management in Agent-based Distributed Environments,” in *Cooperative Information Agents III*, vol. 1652, *Lecture Notes in Artificial Intelligence*, M. Klusch, O. Shehory, and G. Weiss, Eds. Berlin, et al.: Springer-Verlag, 1999, pp. 50-74.

[15] A. Motro and P. Smets, “Uncertainty Management in Information Systems: from Needs to Solutions,” . Norwall, MA: Kluwer Academic Publishers, 1996, pp. 480.

[16] A. Motro, “Multiplex: A Formal Model for Multidatabases and Its Implementation,” ISSE Department, George Mason University, Fairfax, VA, Technical Report ISSE-TR-95-10, 1995.

[17] I. Imam and L. Kerschberg, “Adaptive Intelligent Agents,” *Journal of Intelligent Information Systems*, vol. 9, pp. 211-214, 1997.

[18] D. Rus, R. Gray, and D. Kotz, “Transportable Information Agents,” *Journal of Intelligent Information Systems*, vol. 9, pp. 215-238, 1997.

[19] J. E. Laird, D. J. Pearson, and S. B. Huffman, “Knowledge-Directed Adaptation in Multi-Level Agents,” *Journal of Intelligent Information Systems*, vol. 9, pp. 261-276, 1997.

[20] K. S. Decker and K. Sycara, “Intelligent Adaptive Information Agents,” *Journal of Intelligent Information Systems*, vol. 9, pp. 239-260, 1997.

[21] B. R. Gaines, “Knowledge Management in Societies of Intelligent Adaptive Agents,” *Journal of Intelligent Information Systems*, vol. 9, pp. 277-298, 1997.

[22] A. Kemper, P. C. Lockemann, G. Moerkotte, and H.-D. Walter, “Autonomous Objects: A Natural Model for Complex Applications,” *Journal of Intelligent Information Systems*, vol. 3, pp. 113-150, 1994.

[23] J. P. Yoon and L. Kerschberg, “Query-Initiated Discovery of Interesting Association Rules,” in *Discovery Science: First International Conference, DS'98,* vol. LNAI 1532, *Lecture Notes in Artificial Intelligence*, S. Arikawa and H. Motoda, Eds. Fukuoka, Japan: Springer, 1998, pp. 232-243.

[24] J. P. Yoon and L. Kerschberg, “A Framework for Knowledge Discovery and Evolution in Databases,” *IEEE Transactions on Knowledge and Data Engineering*, 1993.

[25] J. P. Yoon and L. Kerschberg, “A Framework for Constraint Management in Object-Oriented Databases,” presented at International Conference on Information and Knowledge Management, Baltimore, MD, 1992.

[26] W. D. Potter 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, Eds. Amsterdam: North Holland, 1988.

[27] L. Kerschberg, “Expert Database Systems: Proceedings from the First International Workshop,” Menlo Park, CA: Benjamin/Cummings, 1986, pp. 701.

[28] L. Kerschberg, “Expert Database Systems: Proceedings from the First International Conference,” Menlo Park, CA: Benjamin/Cummings, 1987, pp. 501.

[29] L. Kerschberg, "Expert Database Systems: Proceedings from the Second International Conference," Redwood City, CA: Benjamin/Cummings, 1988, pp. 777.

[30] L. Kerschberg, "Expert Database Systems: Knowledge/Data Management Environments for Intelligent Information Systems," *Information Systems*, vol. 15, pp. 151-160, 1990.

[31] G. Wiederhold, "The Roles of Artificial Intelligence in Information Systems," *Journal of Intelligent Information Systems*, vol. 1, pp. 35-56, 1992.

[32] G. Wiederhold, "Mediators in the Architecture of Future Information Systems," *IEEE Computer*, vol. 25, 1992.

[33] C. M. Bowman, P. B. Danzig, D. R. Hardy, U. Manber, M. F. Schwartz, and D. P. Wessels, "Harvest: A Scalable, Customizable Discovery and Access System," University of Colorado, Boulder, Technical Report CU-CS-732-94, March 1995.

[34] C. M. Bowman, P. B. Danzig, U. Manber, and M. F. Schwartz, "Scalable Internet Resource Discovery: Research Problems and Approaches," *Communications of the ACM*, vol. 37, pp. 98—107, 1994.

[35] L. Kerschberg, H. Gomaa, D. A. Menascé, and J. P. Yoon, "Data and Information Architectures for Large-Scale Distributed Data Intensive Information Systems," presented at Proc. of the Eighth IEEE International Conference on Scientific and Statistical Database Management, Stockholm, Sweden, 1996.

[36] D. Weishar and L. Kerschberg, "Data/Knowledge Packets as a Means of Supporting Semantic Heterogeneity in Multidatabase Systems," in *ACM SIGMOD Record*, 1991.

[37] D. Weishar, "A Knowledge-Based Architecture for Query Formulation and Processing in Federated Heterogeneous Databases," in *PhD in Information Technology*. Fairfax, VA: George Mason University, 1993, pp. 230.

[38] G. Wiederhold, S. Jajodia, and W. Litwin, "Integrating Temporal Data in a Heterogeneous Environment," in *Temporal Databases: Theory, Design, and Implementation*, A. U. Tansel, S. Jajodia, and others, Eds.: Benjamin/Cummings, 1993, pp. 563-579.

[39] G. Wiederhold, S. Jajodia, and W. Litwin, "Dealing with Granularity of Time in Temporal Databases," in *Lecture Notes in Computer Science*, vol. 498, R. Anderson and others, Eds.: Springer-Verlag, 1991, pp. 124-140.

[40] X. S. Wang, S. Jajodia, and V. S. Subrahmanian, "Temporal Modules: An Approach toward Federated Temporal Databases," *Information Sciences*, vol. 82, pp. 103-128, 1995.

[41] X. S. Wang, S. Jajodia, and V. S. Subrahmanian, "Temporal Modules: An Approach Toward Federated Temporal Databases," presented at ACM SIGMOD International Conference on Management of Data, Washington, D.C., 1993.

[42] C. Bettini, X. S. Wang, E. Bertino, and S. Jajodia, "Semantic Assumptions and Query Evaluation in Temporal Databases," presented at ACM SIGMOD International Conference on Management of Data, San Jose, CA, 1995.

[43] J. Widom and S. Ceri, "Active Database Systems: Triggers and Rules for Advanced Database Processing," : Morgan Kaufmann Publishers, Inc., 1995.

[44] S. Chakravarthy and J. Widom, "Foreword to Special Issue on Active Database Systems," *Journal of Intelligent Information Systems*, vol. 7, pp. 64, 1996.

[45] U. Dayal, B. Blaustein, A. Buchmann, U. Chakravarthy, R. L. M. Hsu, D. McCarthy, A. Rosenthal, S. Sarin, M. Carey, M. Livny, and R. Jauhari, "The HiPAC Project: Combining Active Databases and Timing Constraints," in *ACM SIGMOD Record*, vol. 17, 1988, pp. 51-70.

[46] U. Dayal, A. Buchmann, and D. McCarthy, "Rules are Objects Too: A Knowledge Model for an Active,Object-Oriented Database System," presented at Proceedings of the Second International Workshop on Object-Oriented Database Systems, Bad Munster am Stein-Ebernberg, Germany, 1988.

[47] R. Alonso, D. Barbara, and H. Garcia-Molina, "Data Caching Issues in an Information Retrieval System," *ACM Transactions on Database Systems*, vol. 15, 1990.

[48] L. Seligman and L. Kerschberg, "An Active Database Approach to Consistency Management in Heterogeneous Data- and Knowledge-based Systems," *International Journal of Cooperative and Intelligent Systems*, vol. 2, 1993.

[49] H. Gomaa and G. K. Farrukh, "An Approach for Generating Executable Distributed Applications from Reusable Software Architectures," presented at IEEE International Conference on Engineering of Complex Computer Systems, Montreal, Canada, 1996.

[50] H. Gomaa, D. Menascé, and L. Kerschberg, “A Software Architectural Design Method for Large-Scale Distributed Information Systems,” *Journal of Distributed Systems Engineering*, 1996.

[51] D. A. Menascé, H. Gomaa, and L. Kerschberg, “A Performance-Oriented Design Methodology for Large-Scale Distributed Data Intensive Information Systems,” presented at First IEEE International Conference on Engineering of Complex Computer Systems, Florida, 1995.

[52] A. Motro, “A Formal Framework for Integrating Inconsistent Answers from Multiple Information Sources,” Department of Information and Software Systems Engineering, George Mason University, Fairfax, VA, Technical Report ISSE-TR-93-106, October 1993.

[53] A. Motro and I. Rakov, “Estimating the Quality of Databases,” in *Proceedings of FQAS 98 Third International Conference on Flexible Query Answering Systems, Lecture Notes in Artificial Intelligence*, vol. 1495, T. Andreasen, H. Christiansen, and H. L. Larsen, Eds. Berlin: Springer-Verlag, 1998, pp. 298-307.

[54] A. Motro and I. Rakov, “Estimating the Quality of Data in Relational Databases,” presented at 1996 Conference on Information Quality, Cambridge, MA, 1996.

[55] A. Motro and I. Rakov, “Not All Answers Are Equally Good Estimating the Quality of Database Answers,” in *Flexible Query-Answering Systems*, T. Andreasen, H. Christiansen, and H. L. Larsen, Eds.: Kluwer Academic Publishers, 1997, pp. 1-21.

[56] D. Weishar and L. Kerschberg, “An Intelligent Heterogeneous Autonomous Database Architecture for Semantic Heterogeneity Support,” presented at IEEE International Workshop on Interoperability in Multidatabase Systems, Kyoto, Japan, 1991.

[57] W. D. Potter, K. J. Kochut, J. A. Miller, V. P. Gandham, and R. V. Polamraju, “The Evolution of the Knowledge/Data Model,” *International Journal of Expert Systems*, vol. 6, pp. 39-81, 1993.

[58] W. D. Potter and R. Trueblood, “Traditional, Semantic and Hyper-Semantic Approaches to Data Modeling,” in *IEEE Computer*, vol. 21, 1988, pp. 53-63.

[59] J. A. Miller, W. D. Potter, K. J. Kochut, A. A. Keskin, and E. Ucar, “The Active KDL Object-Oriented Database System and Its Application to Simulation Support,” *International Journal of Object-Oriented Programming*, vol. 4, pp. 30-45, 1991.