Executive Information Systems, Inc.

Architectural Evolution in DataWarehousing and Distributed Knowledge Management Architecture

By

Joseph M. Firestone, Ph.D.

White Paper No. Eleven

July 1, 1998

Introduction

The Dynamic Integration Problem (DIP) is the problem of proactively and automatically monitoring and managing evolutionary change in data warehousing systems without imposing a traditional and constraining "Top-Down" architecture. It is the problem of providing managers of both data warehouses and data marts, the capability to innovate while still maintaining the integration and consistency of the system. It is increasing recognition of the need for this capability that drives architectural evolution in the DW system. No data warehousing vendor is currently offering a solution to the DIP, though some offer change management through intentional DBA action backed by extensive monitoring and reporting capabilities.

The DIP is now more difficult to solve because data warehousing is increasingly a complex systems integration problem. A full-blown Data Warehousing System may encompass

- the following database servers:
- The data warehouse;
- various data marts (department, function, or application-specific DSSs, using ROLAP, Multi-dimensional, or Column-based Servers);
- One or more Operational Data Stores (ODSs);
- One or more Data Staging Areas; and
- the following application servers:
- Web Servers;
- ETML Servers;
- Data Mining servers;
- Stateless Transaction Servers (e.g., MTS, Jaguar CTS, etc.;
- Business Process engines (e.g. Persistence Power-Tier);
- Document Servers;
- ROLAP Support Servers (e.g., Microstrategy, Information Advantage);
- Report Servers;
- and various front-end OLAP and reporting tools.

The Dynamic Integration problem in this context is three-fold:

- First, an integrated view of all server-based assets is needed;
- Second, flows of data, information, and knowledge throughout this system need to be managed to maintain the common view in the face of change in form and content, and to distribute the system's data, information, and knowledge bases as required, and
- Third, such management needs to occur automatically and without centralizing the system so that the authority and

responsibility for adding new data and information to the system is distributed.

This paper is concerned with DSS/data warehouse system architectural evolution in response to the growing complexity of the enterprise DSS environment, and with the relationship of new architectures to a developing capability to handle the DIP. The paper briefly describes and analyzes the following architectures:

- Top-Down Architecture ;
- Bottom-Up Architecture;
- Enterprise Data Mart Architecture;
- Data Stage/Data Mart Architecture (DS/DMA);
- Distributed Data Warehouse/Data Mart Architecture (DDW/DMA);
- Distributed Knowledge Management Architecture (DKMA);
- Variations with introduction of the ODS.

In addition it comments on the relationship between DKM architecture and data mining, and provides some brief comments on software tools for implementing DKMA.

Top-Down Architecture

Introduced by Bill Inmon [1], this is the first data warehousing architecture. It is represented in Figure One.

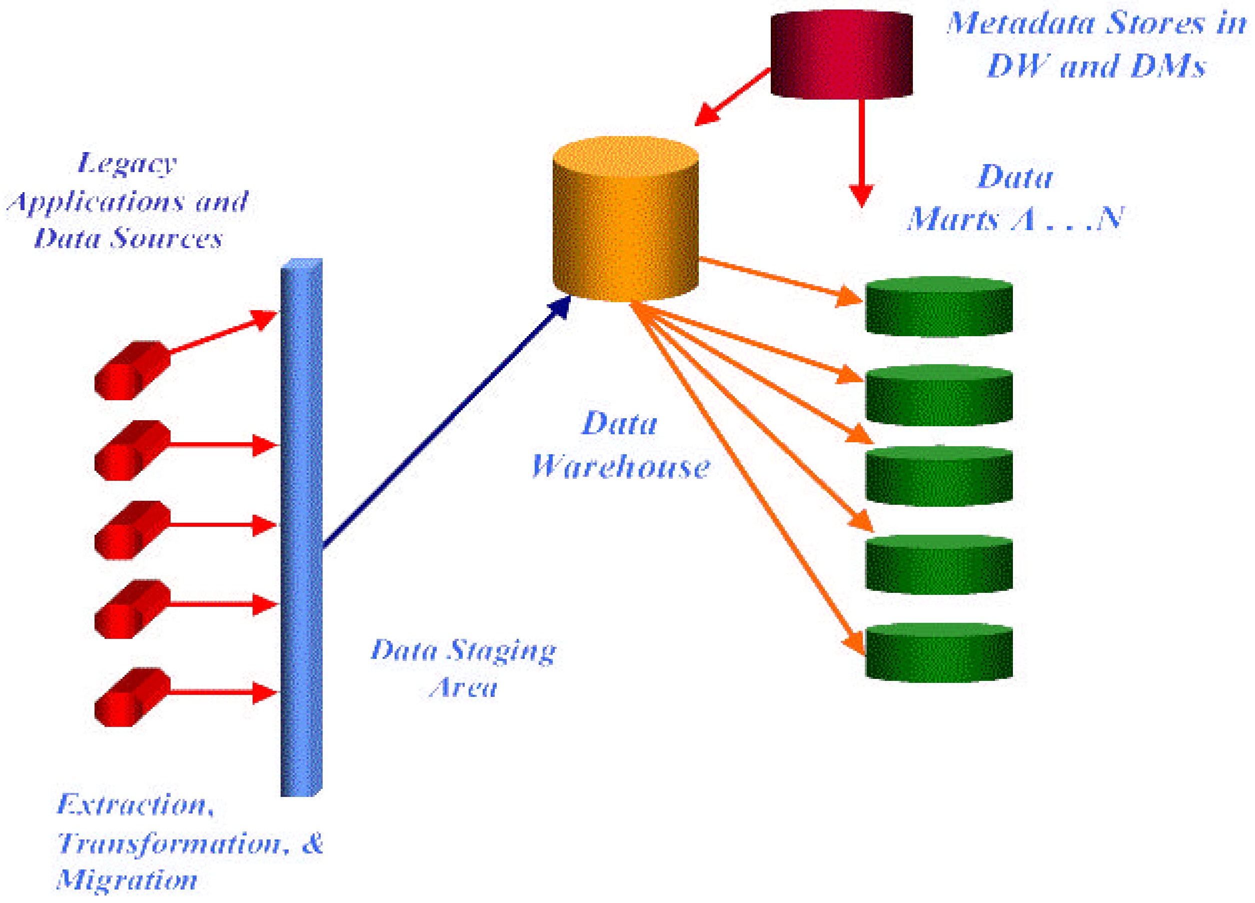

Metadata Stores in DW and DMs
Legacy Applications and Data Sources
Data Marts A . . .N
Data Warehouse
Data Staging Area
Extraction, Transformation, & Migration

Figure One -- Top-Down Architecture

The interaction associated with the architecture begins with an Extraction, Transformation, Migration, and Loading (ETML) process working from legacy and/or external data sources. Extraction transformation, and migration, process data from these sources and output it to a centralized Data Staging Area. Following this, data and metadata are loaded into the Enterprise Data Warehouse and the centralized metadata repository. Once these are constituted, Data Marts are created from summarized data warehouse data and metadata. The data warehouse has an atomic data layer and also contains detailed historical data. In contrast, the data marts contain lightly and highly summarized data and also metadata.

In the top-down model, data warehouses use Normalized E-R Data Models. In contrast, data marts use star schema data models to improve understandability and performance [2].

In the top-down model, integration between the data warehouse and the data marts is automatic as long as the discipline of constituting data marts as subsets of the data warehouse is maintained. Tools (such as Microstrategy's DSS Server) exist to generate data marts from the data warehouse "by pushing a button."

Bottom-Up Architecture

The second data warehousing systems architecture, the "Bottom-up" architecture became popular because the Top-down architecture took too long to implement, was often politically unacceptable, and was too expensive. The Bottom-up architecture also provided a rationalization for department heads and others with budgets to use the new technology of data warehousing to produce application specific DSSs relevant to their organizational roles. Figure Two depicts the Bottom-up idea.

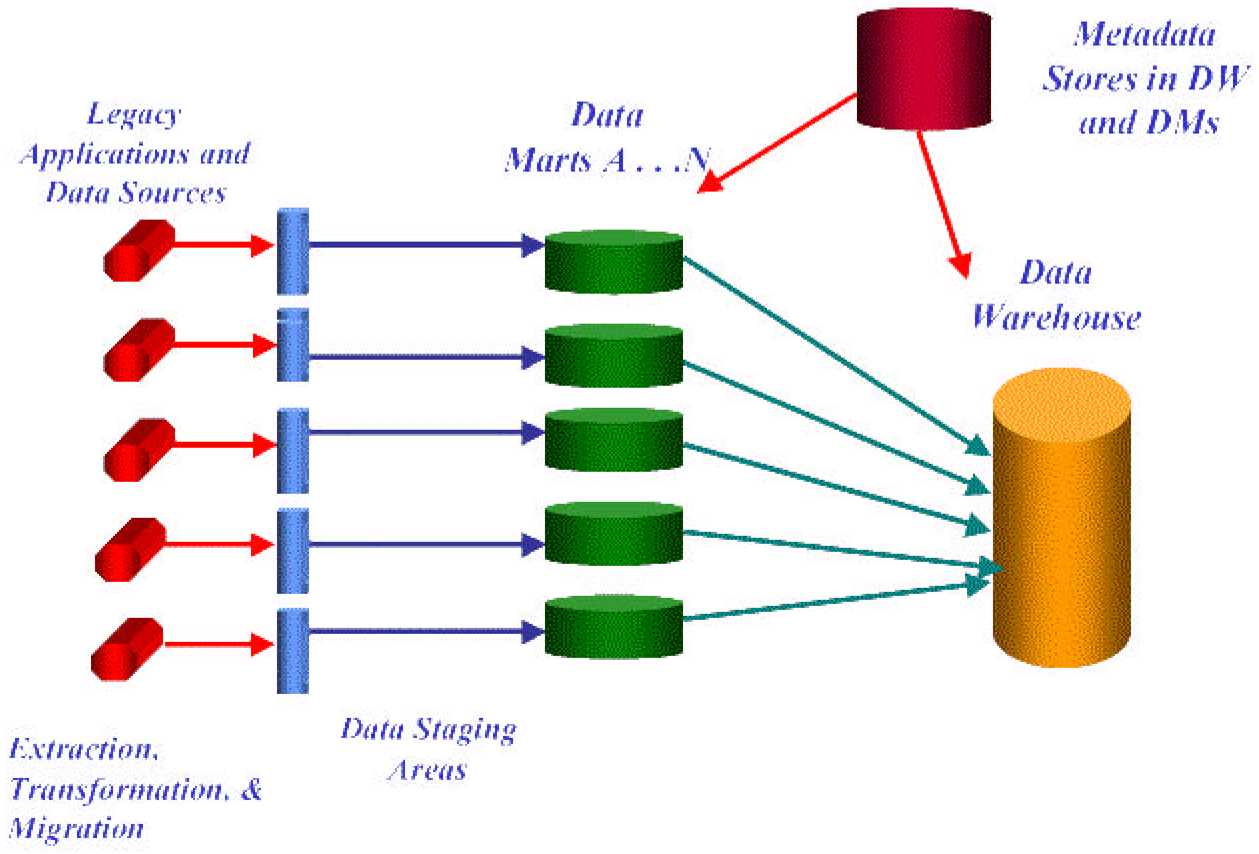

Metadata Stores in DW and DMs
Legacy Applications and Data Sources
Data Marts A . . .N
Data Warehouse
Extraction, Transformation, & Migration
Data Staging Areas

Figure Two -- Bottom-Up Architecture

The central idea in Bottom-up architecture is to construct the data warehouse incrementally over time from independently developed data marts. The process begins with ETML for one or more data marts. No common data staging area is required. There is generally a separate area for each data mart. There may not even be standardization on the ETML tool.

The data marts generally do not use normalized E-R data models. It is generally recognized that when data marts use relational data bases, they should employ dimensional star schema data models, or variants of them, to achieve better performance and understandability. Many data marts don't even use relational technology, but may employ multidimensional database servers or column-based databases (Sybase IQ, and Broadbase).

In Top-down architecture, data marts use lightly and highly summarized data. But in Bottom-up architecture, they also use atomic, and detailed, including historical data. Since the data marts are to be the building blocks of the data warehouse, they must contain all of the data that will appear in the projected data warehouse.

Bottom-up differs from Top-down architecture also, in that it provides no common metadata components across data marts. This is the most important difference between the two architectures from the standpoint of integration. In fact, the evolution of architecture beyond the basic Top-down and Bottom-up patterns is largely the evolution of increasingly sophisticated metadata and meta-object structures in an effort to achieve integration.

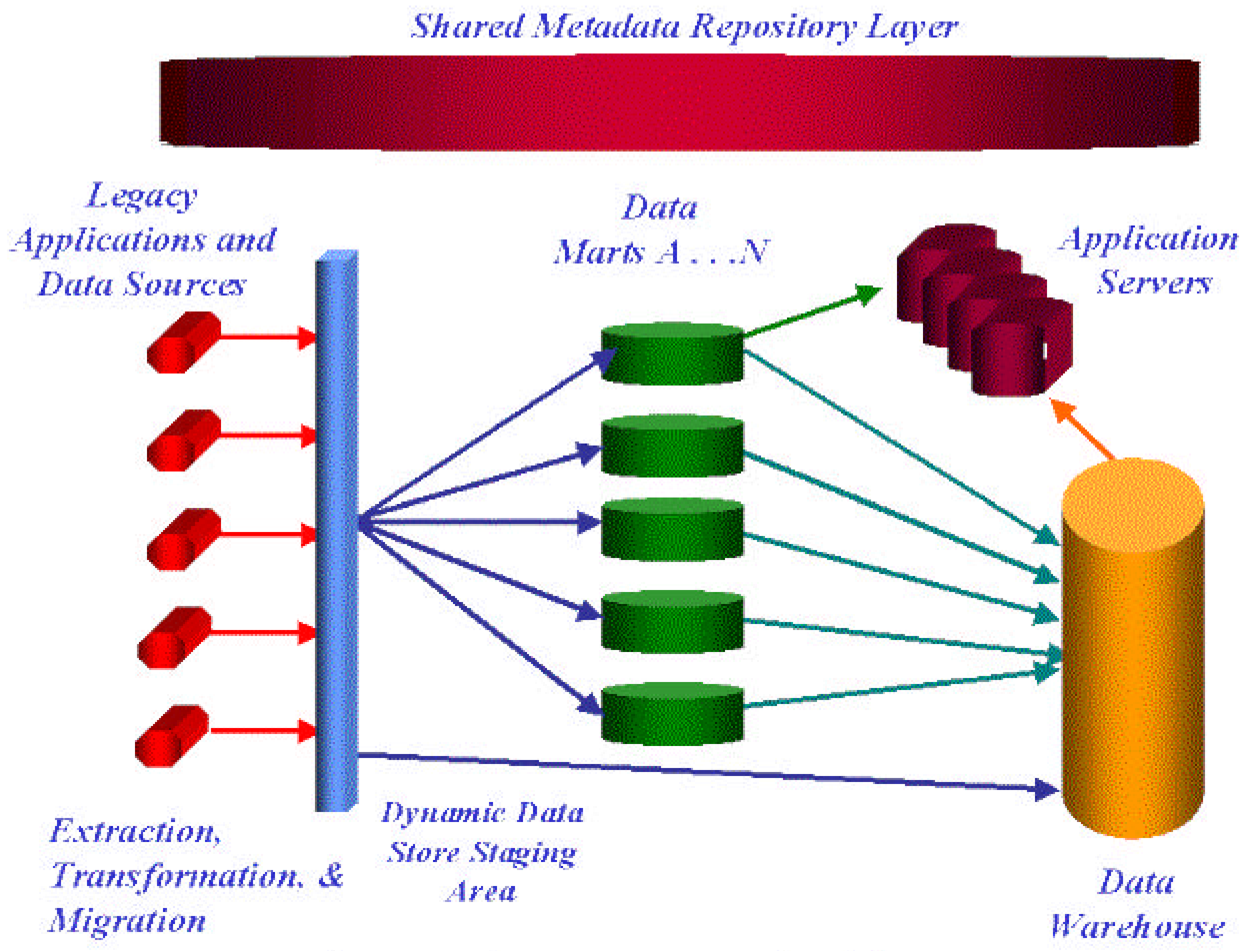
Data warehousing architectures tend to be associated with specific ETML tool vendors, more than they are with other classes of data warehousing/DSS tool vendors. Top-down architecture has been associated with Prism Solutions, Evolutionary Technologies (ETI), and Carleton. Bottom-up architecture was initially associated with "second generation" ETML tool vendors Informatica, Sagent, and Ardent Software (formerly VMark). These vendors have always emphasized the importance of metadata, but initially, at least, their tools did not provide metadata tools for integration across data marts.

While Bottom-up architecture was quite successful in meeting initial expectations in building data marts, it very soon was widely perceived as unacceptable for the long term for the very reason that it failed to provide a common metadata component. Without shared metadata, it is difficult to construct the data warehouse from data marts. So, the Bottom-up architecture, in its pure form fails to fulfill its promise of an incremental approach to the data warehouse. This failure also leads to new "stovepipes" or "legamarts" over time.

<u>Enterprise Data Mart Architecture (EDMA)</u>

Though the "legamart" critique of the pure Bottom-up architecture was decisive, the idea of an incremental approach to data warehouse construction through application specific data marts that deliver value along the road to the comprehensive data warehouse, has real legs. So, Bottom-up supporters quickly modified their approach and their technology to save the idea.

All of the remaining architectures to be discussed here are recent reactions to the need to make an incremental, relatively inexpensive, value-driven approach to data warehousing work [3]. The Enterprise Data Mart Architecture (EDMA), to be discussed in this section [4] is one evolutionary response of Bottom-up supporters to the "legamart" argument. EDMA is illustrated in Figure Three.


Shared Metadata Repository Layer
Legacy Applications and Data Sources
Data Marts A . . . N
Application Servers
Extraction, Transformation, & Migration
Dynamic Data Store Staging Area
Data Warehouse

EDMA supports an incremental approach to the data warehouse through data mart development by creating a shared framework for development. The EDMA framework [5] includes enterprise subject areas, common dimensions, metrics, business rules, and data sources, all represented in a logically common (but not necessarily physically centralized) Global Metadata Repository (GMR). This common framework is established before the EDMA-guided incremental process of data mart/data warehouse development occurs. As development occurs, EDMA is incrementally modified as the development process gradually evolves the foundation for the data warehouse.

Central to the architecture also, is a common data staging area called a Dynamic Data Store (DDS) [6] for extraction, transformation, and migration results. A DDS stores, cleans, and transforms data extracted from operational systems, and also prepares the data for loading into DSS data stores. A DDS is not the same as an Operational Data Store (ODS), as it prohibits DSS processing. The DDS is dynamic in the sense that it is frequently changing as new data is added. The process of establishing and maintaining the DDS also contributes metadata to the GMR. The fact that the architecture incorporates a logically unified data staging area is instrumental in supporting the integration across data marts and with the eventual data warehouse. Because the unified data staging area, along with the GMR, and local data mart metadata repositories, all help to create and maintain semantic consistency in data.

As in the Bottom-up architecture, Data Marts may use multiple data storage technologies as appropriate, but if relational technology is used star schema modeling is the preferred choice.

EDMA is best described by Douglas Hackney in his Understanding and implementing Successful Data Marts [7], and he is the data warehousing practitioner most closely associated with the architecture. Informatica [8] was the first vendor to implement this architecture through its PowerCenter Tool, which implements a DDS, and also a GMR, which it refers to as the Global Data Mart Repository (GDR).

EDMA as implemented through Informatica PowerCenter supports metadata management through extensive monitoring and reporting mechanisms; but not through an automated process. Thus, EDMA does not yet provide a solution to the Dynamic Integration Problem (DIP), though it certainly makes substantial progress towards that goal. In addition to Informatica, EDMA (though the name is not used) is also supported by Carleton through its combined Enterprise Integrator and Passport products (http://www.carleton.com).

Both Informatica and Carleton make use of Object Technology in their metadata repositories. Like Informatica, Carleton provides a framework for beginning to deal with the DIP. But it still provides no automatic mechanisms for synchronizing incremental data marts and the data warehouse.

Data Stage/Data Mart Architecture (DS/DMA)

It seems worthwhile to distinguish DS/DMA from EDMA on one side, and DDW/DMA on the other, since its central idea seems to be spreading. DS/DMA is the same as EDMA with the important exception that no physical enterprise-wide data warehouse is implemented. Instead, the data warehouse is viewed as the conjunction of the data marts in the context of an EDMA-like metadata repository. Figure Four illustrates DS/DMA.

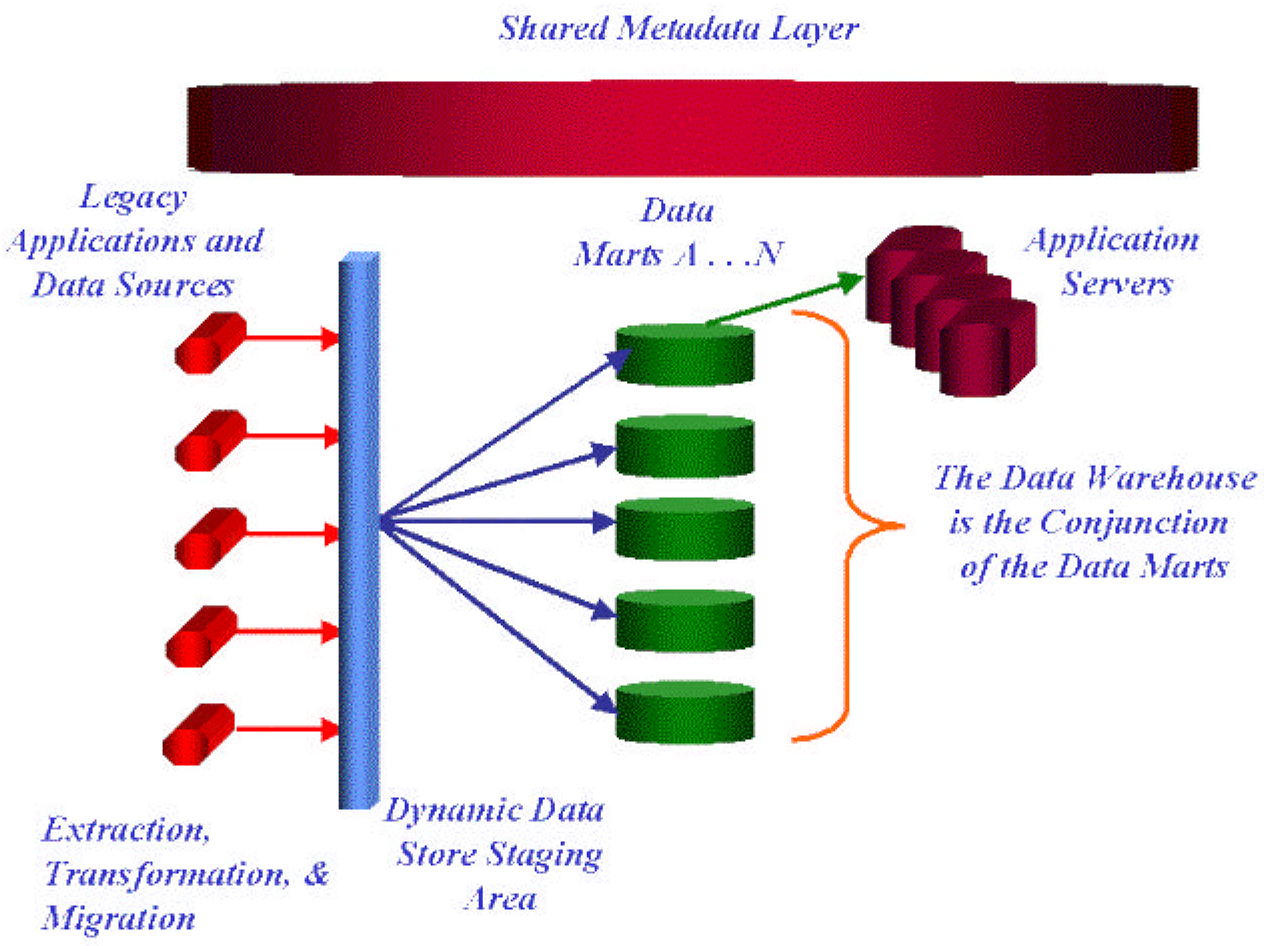

Shared Metadata Layer
Legacy Applications and Data Sources
Data Marts A . . .N
Application Servers
The Data Warehouse is the Conjunction of the Data Marts
Extraction, Transformation, & Migration
Dynamic Data Store Staging Area

Figure Four -- DS/DM Architecture

The repository provides a common view of DSS resources across data marts, but not necessarily any data or tables that represent a view of global enterprise characteristics. There is no guarantee that the conjunction of subject matter, department, and/or application specific data marts will provide access to such global attributes, as would a data warehouse.

In other words, the view that the data warehouse is just the logical conjunction of the data marts, ignores the fact that data marts are constructed to satisfy the interests of actors whose departmental locations, or less than global responsibilities, bias them toward warehousing data that may not measure or describe global enterprise properties. That is, their subject matter, departmental, and application specificity will not deal with enterprise wide interests.

The conjunction of such data marts may allow one to derive aggregate properties that are at the enterprise level of analysis, and even structural properties describing relationships among departments or groups or individuals in the enterprise. But they will not allow one to derive global properties of the enterprise, because these are emergent characteristics of organizational interaction. They cannot be derived from aggregations or structural measurements on enterprise components. They must be measured through data stores that directly address global enterprise matters.

This argument assumes that data marts are never concerned with global enterprise matters, and this assumption may seem unreasonable since such things as sales data marts may clearly involve such properties as total enterprise sales revenue. But the terms data warehouse and data mart are often used loosely, and just as there are sales data marts, there are also sales data warehouses. So what's the distinction between these two concepts?

I think there is none, and that this looseness of language is the real explanation for the emergence of DS/DMA. If we accept a definition of data mart that allows us to apply the term to enterprise wide, application specific DSS data stores; then the central idea of DS/DMA, that the global data warehouse is the conjunction of all data marts, is much more plausible. Application specific, enterprise wide data marts will contain global characteristics of the enterprise, and the conjunction of them may contain all such characteristics.

But, in the end this definition of data mart, even though it sometimes accords with common usage, is not reasonable.

There is really no need to stretch the data mart concept to encompass global phenomena. It is much easier to maintain that the essence of the data warehouse/data mart distinction is the global/local distinction, and therefore that sales data marts, if global in nature, are really application (or at least subject matter) specific, sales data warehouses. That is, we should recognize data warehouses and data marts, and within the data warehouse category "galactic" data warehouses, and application or subject matter specific ones.

Primary tool providers for DS/DMA are again Informatica and Carleton. Both tools emphasize the importance of shared metadata arising from every stage of the data warehousing process. Both tools emphasize the importance of a GMR, and lastly, both tools emphasize the importance of a centralized data staging area in producing data consistency and integrity.

<u>Distributed Data Warehouse/Data Mart Architecture (DDW/DMA)</u>

DDW/DMA is also similar to EDMA (See Figure Five). Like EDMA it provides a dynamic data staging area and a common view of metadata across the enterprise in the form of a shared metadata repository. The distinctive characteristics of DDW/DMA are two.

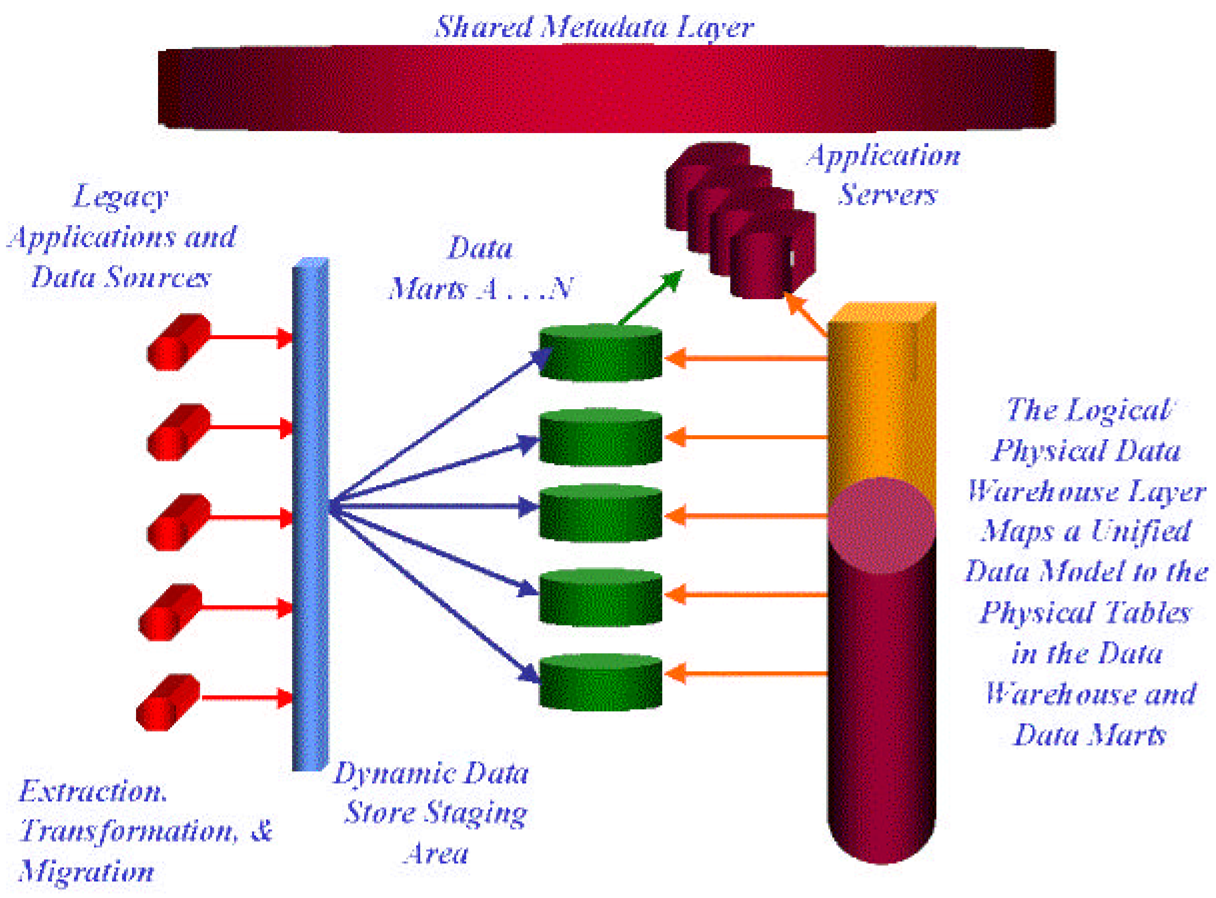

Shared Metadata Layer
Application Servers
Legacy Applications and Data Sources
Data Marts A . . . N
The Logical/ Physical Data Warehouse Layer Maps a Unified Data Model to the Physical Tables in the Data Warehouse and Data Marts
Extraction, Transformation, & Migration
Dynamic Data Store Staging Area

Figure Five -- DDW/DM Architecture

First, it provides a logical database layer mapping a unified logical data model to physical tables in the various data marts. And second it provides transparent querying of the unified logical database across data marts and data warehouses along with caching and joining services. Thus, the distributed character of the data warehouse/data mart system is made transparent to users.

Leading tool providers supporting this architecture are Informatica, Carleton, and Sybase Adaptive Server. These tools are all offered as part of Sybase's Warehouse Studio [9]. Informatica and Carleton provide the unified logical view of the data warehouse, and Sybase Adaptive Server provides the ability to query, cache, and join across data marts and data warehouses as necessary. HP Intelligent Warehouse is also an example of this architecture. But it is a product in transition following its recent purchase by Platinum Technologies.

This is the most adaptable of the architectures discussed to this point, but it still reflects the limitations of the relational viewpoint when it comes to handling objects and processes, and it still doesn't support distributed and automated change capture and management.

Distributed Knowledge Management Architecture (DKMA)

DKM architecture is an evolving O-O/Component-based architecture applicable to enterprise wide systems incorporating multiple processing styles including DSS, OLTP, and Batch processing. These systems are called Distributed Knowledge Management Systems (DKMS), a concept I've introduced in previous work [10]. Here DKM architecture is applied to data warehouse/data mart -based DSS systems.

Top - Down and Bottom-Up architectures may be viewed as two-tier architectures utilizing clients and local or remote databases. EDMA, DS/DMA, and DDW/DMA may be viewed as adding middleware and tuple [11] layers to earlier architectures to provide the capability to manage warehouse systems integration through unified logical views, monitoring, reporting, and intentional DBA maintenance activity. But tuple-layer based management still doesn’t provide automatic feedback of changes in one component of a data warehousing system to others.

DKM architecture may be viewed as adding an object layer to EDMA or to DDW/DMA to provide integration through automated change capture and management. Figure Six depicts DKM architecture.

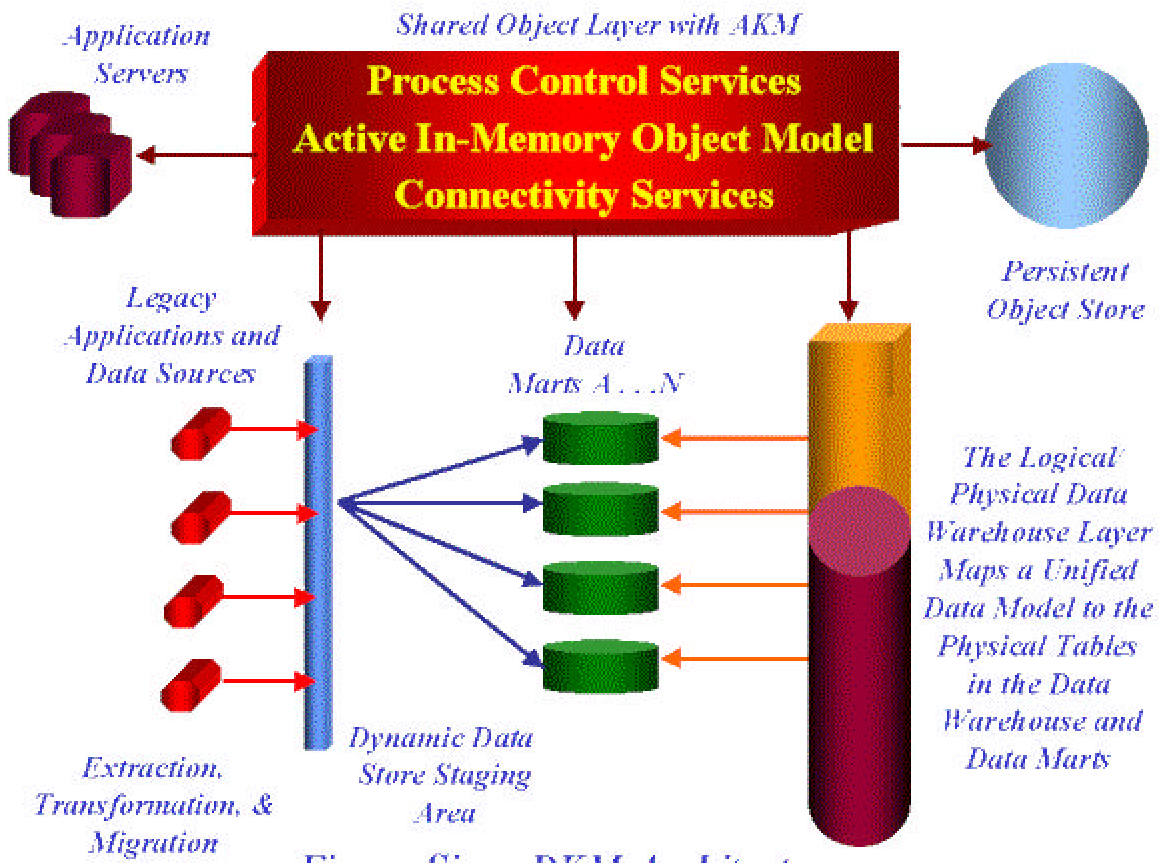

Application Servers
Shared Object Layer with AKM
Process Control Services
Active In-Memory Object Model
Connectivity Services
Persistent Object Store
Legacy Applications and Data Sources
Data Marts A . . . N
The Logical/ Physical Data Warehouse Layer Maps a Unified Data Model to the Physical Tables in the Data Warehouse and Data Marts
Dynamic Data Store Staging Area
Extraction, Transformation, & Migration

Figure Six -- DKM Architecture

The object layer contains an architectural component called an Active Knowledge Manager (AKM) [12]. The AKM provides process control/distribution services, an in-memory active object model accompanied by a persistent object store, and connectivity to a variety of data store and application types.

Process Control Services include:

- in - memory proactive object state management and synchronization across distributed objects;
- component management;
- workflow management;
- transactional multithreading.

The in-memory Active Object Model and Persistent Object Store Model components of the AKM include:

- Event-driven behavior;
- DKMS-wide model with shared representation;

- Declarative business rules;
- Caching through partial instantiation; and
- A Persistent Object Store for the AKM.

Connectivity Services of the AKM include:

- Language APIs: C, C++, Java, CORBA, COM
- Databases: Relational, ODBC, OODBMS, hierarchical, network, flat file, etc.
- Wrapper connectivity for application software: custom, CORBA, or COM-based.
- Applications including all categories mentioned in earlier discussion of the Dynamic Integration problem whether these are mainframe, server, or desktop - based.

The DKM Architecture and the AKM provide the solution to the Dynamic Integration Problem, because only the DKMA among the preceding architectures, supports distributed, proactive monitoring and management of change in the web of data warehouse, data mart, web information servers, component transaction servers, data mining servers, ETML servers, other application servers, and front-end applications comprising today's Enterprise DSS/Data Warehousing System.

Variations With Introduction of the ODS

Each of the architectures covered may vary with the addition of an Operational Data Store (ODS). According to Inmon, Imhoff, and Sousa, [13]: "An ODS is a collection of data containing detailed data for the purpose of satisfying the collective, integrated operational needs of the corporation . . . The ODS is:

- subject-oriented,
- integrated,
- volatile,
- current-valued,
- detailed."

The ODS is like a data warehouse in its first two characteristics, but it is like an OLTP system in its last three characteristics. Its purpose is to support operational, tactical decisions. The workload of an ODS involves four kinds of processing: loading data, updating, access processing, and DSS-style analysis across many records [14].

The four types of ODS processing are the source of difficulties in optimizing ODS processing. It is difficult to optimize performance over all four types.

Look at the above architectures in relation to the ODS. It is clear that an architecture that will support both DSS and OLTP-style processing is needed in order to optimally integrate the ODS into the broader data warehousing architecture. In particular, process control services will be very important for the OLTP-style of processing we find in the ODS. Also, distribution of ODS objects across multiple servers will help ODS performance. Finally, in-memory processing in distributed AKMs can do much to upgrade performance in a distributed ODS. Of course, only one of the above architectures can provide these capabilities for the ODS: the DKM Architecture.

DKM Architecture and Data Mining

A key emerging capability in DKMS and data warehousing systems is Knowledge Discovery in Databases (KDD) or Data Mining [15][16]. The key mechanism for KDD is the data mining server.

Here are some difficulties with current data mining server products:

- It's difficult to incorporate new data mining algorithms, and therefore keep pace with new developments coming out of the research world;
- Many products require that data must be transported to proprietary data stores before data mining can occur;
- Models produced by the data mining algorithms are not freely available to power users unless they use the data mining tool itself;
- It is difficult to incorporate validation criteria not initially incorporated in the data mining tool into the KDD process;
- There are few "open architecture" commercial data mining tools.

To solve these problems a product class called An Analytical Data Mining Workbench (ADMW) should be developed. The ADMW needs:

- Easy and convenient encapsulation of new algorithms into object model classes;

- Capability to mine data from any data source in the enterprise;
- Incorporation of analytical models into an object model repository;
- A modifiable validation model,
- Integration of legacy data mining applications with the ADMW.

An ADMW with these capabilities would meet all of the difficulties specified above.

There are a number of reasons why DKM Architecture can help in developing an ADMW product. First, new algorithms can easily be encapsulated in objects through the wrapping capabilities of the AKM [17]. These wrapping capabilities can be used to create standard object interfaces for objects incorporating new data mining algorithms developed in universities and research centers. In turn, the standard interfaces can be used to plug the new objects including their new methods into the analytical workbench. The time to market of new data mining algorithms could be substantially reduced.

Second, persistent data can be brought into the AKM’s in-memory object model for data mining without first relocating it from its current data mining data mart relational data store [18]. AKM's connectivity allows it to access relational tables to read data in "chunks" for high speed processing. To do this efficiently it is necessary to maintain relational production data mining tables side-by-side with the dimensionally modeled relational tables of the data mart. The function of the data mining tables is to maintain data in optimal form for input into an in-memory data mining engine that would be integrated with the AKM.

Third, data mining can be performed by executing the analytical models in memory. The AKM's connectivity allows it to integrate the data mining engine with its process control services and in-memory object model.

Fourth, analytical models produced by an AKM -based application would be placed in an object model repository. Any model generated by the data mining engine can be expressed as an object with the model as one of its methods. So analytical models entering the data mining process can be saved and placed in the object model repository.

Fifth, customized validity criteria could be added by modifying the validation model [19] in the repository. A validation model is nothing but another object in the object model. As such, it can be accessed by users of the repository, and it's content can be changed, using the tools made available by the system to edit and reformulate analytical models of other kinds.

Sixth, legacy data mining applications could be integrated using AKM connectivity services. If a language API (C/C++, Java, CORBA, DCOM) exists, AKM will be able to access the legacy application's functions once these are described in the AKM object model. If an API is absent, the AKM component can be used to "wrap" the data mining application and make its functions available to the AKM.

When considering the above points, keep in mind that there is no ADMW with all of the above capabilities at present. Data Mining is a rapidly growing field, but the market niche represented by the ADMW is empty.

<u>DKM Architecture and Software Tools</u>

To implement DKM Architecture in a DKMS you need the full range of tools now used to create data warehousing systems. In addition you need tools specifically for the AKM component. These include:

- An object modeling RAD environment providing extensive process control services and connectivity (e.g. Template Software’s Enterprise Integration Template, Forte, DAMAN's InfoManager, a combination of Ibex's DAWN workflow product along with its Itasca Active Object Database, a combination of Rational Rose, Persistence Power-Tier; and Iona’s Orbix);
- Technology for constructing software agents to proactively monitor components of the DKMS (e.g. CA Unicenter TNG, ObjectSpace’s Voyager, DAMAN's InfoManager, and Persistence Power-Tier).
- An OODBMS to serve as a persistent object repository for the AKM component (ObjectStore, Objectivity/DB, Jasmine, Versant, Itasca).

<u>Conclusion: Coordinated Evolution of the Enterprise DKMS</u>

An enterprise DKMS, or for that matter, an enterprise data warehousing system is not a static construct in which the whole is the sum of the parts. It is, or should be, an intelligent system that learns and adapts over time. In such a system, the parts, or data marts, should not be subsets of the data warehouse. Nor should the data warehouse be the logical union of the data marts. Data marts are, at best fuzzy subsets of data warehouses; and data warehouses represent the fuzzy union of data marts.

The DKM architecture provides for coordination of data mart development in enterprises from the get-go. The vision of the DKM architecture is not top-down coordination, however, but the coordination of components having fuzzy relations to one another.

There are a number of reasons why the relationship between data warehouses and data marts should be fuzzy, but they all stem from consideration of the role of business users in the dynamics of the developing relationships between data marts and the data warehouse within an enterprise. Whether a primarily top down, bottom up, or metadata guided architecture is used, there will always be continuous user feedback in response to data mart and data warehouse activity, because such feedback is part of the learning process in an enterprise.

User requirements are not static. They tend to evolve on exposure to new applications and new technologies as the users learn. Changes in requirements are not limited only to faster hardware, or better techniques for data mining, or improved database software, or GUI interfaces. They're also going to include changes in information, knowledge, and data requirements. New attributes and tables might need to be added to data marts. Old tables might need to be reorganized. New requirements may therefore impact either data or object models at both the data mart and data warehouse levels. New causal dimensions and attributes will be conceived and created by users responding to their new technology. New ETML processes may be needed to process these causal dimensions.

How will the demands for new information and knowledge be handled? Will users be told to wait until a central coordinating team adds new dimensions to a centralized data model? Or will departments supplement their data marts with new data; go through a new, if limited ETML process; and constitute a revised data mart (now a fuzzy subset of the data warehouse) that will solve their specific analysis problem?

If the central coordinating team disapproves adding new dimensions because they've "taken the pledge" to guard the integrity of the set of "master conformed dimensions," will they be able to stop a Vice-President of Marketing from doing his job by adding the new causal dimensions? Should they be able to? And if they are allowed to build their fuzzy subset data marts would this again lead to stovepipes?

Not if the enterprise implements ***dynamic integration*** through continuous feedback from data marts to the data warehouse, and continuous feedback integration of changes that seem necessary at either the data mart or data warehouse levels. If a continuous pattern of adjustment to data mart changes is adopted as policy, a pattern of gradual evolution of data warehouses and data marts will occur, and stovepipes will be avoided.

But data marts will not generally be logical subsets of data warehouses, nor will data warehouses be merely the union of data marts. The pattern of development will involve continuous feedback from the periphery to the center, and continual adjustment of both the periphery and the center to each other. The enterprise data warehouse will not bring a once-and-for-all decision support nirvana, but a much healthier process of continuous conflict, learning, and growth of business intelligence.

The means to reach this state of coordinated evolution is the DKMS. And to implement the DKMS we need an architecture consistent with its purposes and with the requirement that the dynamic integration problem give way to the pattern of coordinated evolution. The DKMA is that architecture.

References

[1] W. H. Inmon, Building the Data Warehouse, 2nd ed. (New York, NY: John Wiley & Sons, 1996)

[2] The clearest statements on dimensional modeling and star schemas, are Ralph Kimball's. See The Data Warehouse Toolkit (New York, NY: John Wiley & Sons, 1996), and "A Dimensional Modeling Manifesto," DBMS, August 1997. There is considerable controversy over whether dimensional modeling or E-R modeling should be used for the data warehouse. But everyone seems to agree that dimensional modeling is the preferred technique for relational data marts. See W. H. Inmon, Claudia Imhoff, and Ryan Sousa, Corporate Information Factory (New York, NY: John Wiley & Sons, 1998), 76-78

[3] I'd like to thank my fellow participants in the dwlist server group (dwlist@datawarehousing.com) for the continuing architectural discussions carried on in the group. Our discussions motivated me to think through this paper on architectural evolution in data warehousing, and I'm sure your friendly response to this paper will contribute to the evolution of my ideas on architectural evolution.

[4] See Douglas Hackney, Understanding and Implementing Successful Data Marts (Reading, MA: Addison-Wesley, 1997), Pp. 52-54, 183-84, 257, 307-309, for clear and more detailed treatment of EDMA. See also Informatica's treatment of the EDMA concept in its "PowerCenter Technical Overview" White Paper, available at http://www.informatica.com.

[5] Hackney, Ibid.

[6] Informatica, Op. Cit., defines DDS as ". . . itself a data mart." This is carrying things a bit too far. If data mart means anything, it means a DSS processing platform. The DDS is not that. Informatica also says that "the DDS maintains a data model that closely resembles that of the operational systems." This may be true in cases where both the operational data and its DSS platform target are in relational form. But data stage processing, as Ralph Kimball has made clear, is overwhelmingly sequential in nature. So except in cases such as the above where it is inconvenient to take the data out of its relational form, the DDS should actually be in flat file form.

[7] Hackney, op. cit.

[8] Informatica, op. cit.

[9] http://www.sybase.com/products/dataware/

[10] I introduced the DKMS concept in two previous White Papers "Object-Oriented Data Warehouse," and "Distributed Knowledge Management Systems: The Next Wave in DSS." Both are available at http://www.dkms.com/White_Papers.htm.

[11] Wolfgang Keller, Christian Mitterbauer, and Klaus Wagner, "Object-Oriented Data Integration," in Mary E. S. Loomis, and Akmal B. Choudri (eds.), Object Databases in Practice (Upper Saddle River, NJ: Prentice-Hall, 1998), pp. 7-11

[12] The ideas for the AKM owe much to the following White Papers. Template Software, "Integration Solutions for the Real-Time Enterprise: EIT - Enterprise Integration Template," Dulles, VA, White Paper May 8, 1998. See also http://www.template.com. Persistence Software, "The PowerTier Server: A Technical Overview" at http://www.persistence.com/products/tech_overview.html, and John Rymer, "Business Process Engines, A New Category of Server Software, Will Burst the Barriers in Distributed Application Performance Engines," Emeryville, CA, Upstream Consulting White Paper, April 7, 1998 at http://www.persistence.com/products/wp_rymer.html. Two other products that could be used to develop the AKM component are DAMAN's InfoManager (inquire at http://www.damanconsulting.com), and Ibex's DAWN workflow product along with its ITASCA active database (at http://www.ibex.ch/)

[13] W. H. Inmon, Claudia Imhoff, and Ryan Sousa, Op. Cit., Pp. 87-88

[14] Ibid. Pp. 95-97

[15] See Usama M. Fayyad, Gregory Piatetsky-Shapiro, Padhraic Smyth, and Ramasmy Uthurusamy (eds.), Advances in Knowledge Discovery and Data Mining (Cambridge, MA: M.I.T. Press, 1996)

[16] A recent white paper of mine "Knowledge Management Metrics Development: A Technical Approach," treats KDD as a use case in the DKMS. See it at http://www.dkms.com/White_Papers.htm.

[17] A good summary of wrapping is in Thomas J. Mowbray and Ron Zahavi, The Essential CORBA: Systems Integration Using Distributed Objects (New York, NY: John Wiley & Sons, 1995), Pp. 232-238.

[18] These architectural views on data mining owe much to the architecture of the Datasage production data mining tool (http://www.datasage.com)

[19] See the development of this idea in my "Knowledge Management Metrics Development . . ." Op. Cit., Pp. 15-18.

Biography

Joseph M. Firestone is an independent Information Technology consultant working in the areas of Decision Support (especially Data Marts and Data Mining), Business Process Reengineering and Database Marketing. He formulated and is developing the idea of Market Systems Reengineering (MSR). In addition, he is developing an integrated data mining approach incorporating a fair comparison methodology for evaluating data mining results. Finally, he is formulating the concept of Distributed Knowledge Management Systems (DKMS) as an organizing framework for the next business "killer app." You can e-mail Joe at eisai@home.com.

[Up] [Data Warehouses and Data Marts: New Definitions and New Conceptions]
[Is Data Staging Relational: A Comment]
[DKMA and The Data Warehouse Bus Architecture]
[The Corporate Information Factory or the Corporate Knowledge Factory]
[Architectural Evolution in Data Warehousing]
[Dimensional Modeling and E-R Modeling in the Data Warehouse]
[Dimensional Object Modeling] [Evaluating OLAP Alternatives]
[Data Mining and KDD: A Shifting Mosaic]
[Data Warehouses and Data Marts: A Dynamic View]
[A Systems Approach to Dimensional Modeling in Data Marts]
[Object-Oriented Data Warehousing]