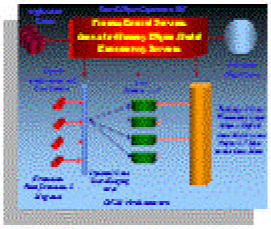

DKMS Briefs

Joseph M. Firestone

DKMS Brief No. Nine: Enterprise Integration, Data Federation, and the DKMS: A Commentary

Stonebraker's Enterprise Integration Solutions

In an eloquent article called "United We Stand," [1] Michael Stonebraker offered a viewpoint on the "islands of information" problem. He defined Enterprise Integration as ". . . the ability to read from and write to all of the applications and data sources across the enterprise." And he indicated: "Such integration supports unified views of information and lets you update synchronously across systems." Stonebraker goes on to distinguish four approaches to the "islands of information" problem: (1) application integration, an approach he associates with a number of commercial Enterprise Application Integration packages; (2) data warehousing systems; (3) messaging systems; and (4) data federation systems.

The ***application integration*** approach implements integration by attaching a layer of "glue" providing an interface from an external integration system to each application being integrated. Stonebraker faults commercial EAI systems for their: (1) failure to provide a unified view of data across databases; (2) inability to handle "conceptual mismatches; and (3) difficulties in scaling to the enterprise level.

Stonebraker views ***data warehousing*** as providing integration of applications at the data level. The integration is physical in that data must be extracted, cleansed, transformed and moved from their original data stores to data warehouses or data marts. He lists the following drawbacks of the data warehousing approach: (1) stale data, only periodically refreshed; (2) inflexibility in analysis due to unavailability of data not in a data warehouse or data mart; (3) copy costs providing a barrier to integration and therefore scalability in certain instances; and (4) overloaded resources because of the strain on the system caused by retail transactional histories.

Messaging systems connect applications by providing an information bus through which updates are published to subscribers. The drawbacks of messaging systems, according to Stonebraker are: (1) subscribers can integrate data only if the data owner has published it onto the information bus, (2) scalability problems, because the bandwidth of the information bus is too narrow to publish all enterprise data, and (3) difficulties in working with dynamic environments.

Data federation performs integration while leaving the source data in place. Multiple physical databases are integrated into a single logical one, and independent enterprise systems produce integrated global behavior through economic forces. Data federation systems retain local control while scaling to hundreds of machines. They also support a global view of enterprise data resources, dynamic load balancing across system resources, and adapt and adjust query execution accordingly.

After stating the four approaches Stonebraker makes three important points:

- "**Data federation systems are coming.** The only way to integrate realtime data in many independent system is to federate the systems. . ."
- "**Data Transformation is a required feature . . ."** , since "there are few semantically identical data sets. . . In a data federation system, conversion is done as the end user accesses data."
- **"You can achieve scalability to the enterprise only by logical integration."** Messaging and data warehousing systems create physical integration in specific data stores. "Restricting integration to a single physical machine will prohibit enterprise level scalability. Only logical integration over multiple machines offers the possibility of enterprise scalability."

And he concludes that an enterprise must decide when to perform application integration and when to perform data integration. And further, that application integration should be used "when performing update integration for small numbers of existing applications; while data integration should be used "when you need unified views or are integrating many systems."

If one does choose the data integration route, then the critical choices are between physical and logical integration, and between aged data and realtime data. Physical integration of aged data leads to data warehouses and data marts, while physical integration of realtime data leads to the Operational Data Store and messaging systems. Logical integration of either aged or realtime data, on the other hand, leads to data federation systems. Logical integration is necessary if it is impractical to move the data to a single location.

Finally, Stonebraker believes that there are powerful market trends moving away from data warehousing and toward data federation. One such trend is marked by business conditions "demanding realtime information, which warehouses can't provide." The second trend is the increasing need of businesses to respond to changing business conditions and external events. This need creates a corresponding need for IT solutions that can produce Dynamic Enterprise integration (DEI) (my term, rather than Stonebraker's), rather than static integration. The combination of these trends suggests a move toward data federation and away from data warehousing, because only data federation can produce the necessary access to both integrated realtime data about rapidly changing conditions, and integrated aged data providing the historical and baseline information, needed for survival and competitive advantage.

Stonebraker's treatment of enterprise integration is one of those possibly seminal articles providing a philosophy that can quickly grab hold of the IT industry and provide direction for a major cycle of activity. As such, it deserves careful examination from readers and critics alike. The issues he raises need a full airing from as many points of view as possible, before we quickly march down the road he has laid before us. The rest of this paper will try to provide this airing and also an alternative point of view – the point of view of Distributed Knowledge Management System-based (DKMS-based) integration.

Natural and Artificial Systems Integration

Enterprise integration is not synonymous with either integrating data or integrating applications, as Stonebraker's viewpoint suggests. Enterprise integration refers to integrating a natural system, a social system, to be more precise; and not to integrating only one subsystem of this broader system -- its artificial computing system.

When we use the unqualified general term loosely, and apply it to artificial systems integration, rather than natural systems integration, we gloss over the fact that there isn't universal agreement about what we mean in this more specific context. We may all agree that we really don't mean enterprise integration as that term is applied to social systems. We also may agree that we all have in mind integration of certain aspects of the computing system of enterprises. But that is where our agreement ends.

Stonebraker distinguishes between:

a. enterprise data integration and
b. enterprise application integration, and also between physical integration and logical integration within the enterprise data integration category.

But one can also distinguish:

c. enterprise artificial information integration, and
d. enterprise artificial knowledge integration,

as major categories of approaches to enterprise artificial systems integration. One's view of what constitutes an adequate enterprise integration solution will depend on which of these types of integration one thinks is essential to the enterprise's ability to compete and adapt.

Stonebraker's view of enterprise artificial systems integration is also too restricted in its development of the application integration category. His discussion of data integration is relatively rich. His discussion of application integration is perfunctory, by comparison.

And, considering the relatively short-shrift he gives application integration, it is not unfair to say that his discussion is biased towards a data integration approach.

This is certainly the type of approach that has been most successful in physically integrating data in recent years. But you can't arrive at a fair assessment of its effectiveness for enterprise artificial systems integration through an analysis that either excludes its main competitors, or provides them with less than comprehensive consideration. Especially since the focus of our integration activity is now shifting from physical integration, which favors data-centric approaches, to logical integration, which does not necessarily favor data.

Enterprise Application Integration (EAI)

Stonebraker criticizes EAI for focusing on synchronized updates across applications and not providing a unified view of data across the enterprise. He also faults it because each application has its own version of some key concept such as "customer" or "order," and he thinks that reconciling such "conceptual mismatches" is much more difficult than integrating the records in an underlying database. Finally, he thinks that the scalability obstacle inherent in integrating 100 different applications by creating 100 specific adapters connecting the applications to the integrative "glue" of an EAI system is insurmountable. These criticisms of EAI are hard to sustain without far more detailed evidence and benchmarking results than Stonebraker presents in his article.

First, certain EAI packages such as Vitria Technology's BusinessWare [2] and Template Software's Enterprise Integration Template (EIT) [3] do provide a unified view of the enterprise. EIT provides a unified view of objects, data, and methods across the enterprise. Vitria provides a unified view of business processes and of data and methods as they relate to processes.

Second, certain EAI packages have the capability to handle conceptual mismatches. They do so by using an object model to provide multiple interpretations of the same data. Again, Template's EIT provides a good example of an existing product that uses a semantic object model to resolve ambiguities across enterprise stovepipes.

Third, Stonebraker's scalability criticism needs to be documented with much more evidence before it can be uncritically accepted. All of the major EAI vendors claim scalability, and many use federation architectures containing distributed application servers to back that claim.

NEON [4] and Vitria Technology, two of the examples cited by Stonebraker, claim scalability through distributed processing and broadly-based connectivity to varieties of data sources. Template, meanwhile, offers as an integrative mechanism a virtual, in-memory, cached, self-reflexive object model, resident in distributed application servers, along with connectivity to data stores and applications of all types.

The above is not to say that Stonebraker is necessarily incorrect in all his claims about the application integration approach. But he seems clearly incorrect in claiming that practitioners of the approach don't generally provide a unified view of the enterprise, and it will take a lot more proof than he presents to support his other two criticisms.

The Data Federation Approach

Stonebraker's various comments on the data federation approach also contain some questionable arguments and conclusions. First, "data federation systems are coming." And it's true that ***federation*** is the way to go in enterprise artificial systems integration. But federation is ***not*** **"**the only way to integrate realtime data in many independent systems," if by this Stonebraker means federating data stores alone. Realtime data can also be integrated by federated systems of databases and application servers through an object layer, as has been amply demonstrated by various EAI vendors.

Second, and while "data transformation is a required feature . . ." , since "there are few semantically identical data sets. . .", it's also true that data transformation is not restricted to data integration approaches. If it is true that "in a data federation system, conversion is done as the end user accesses data," it is also true that in-place, realtime conversion can occur in any of the other major integration approaches, provided only that they employ both federated servers and an object layer for performing integration.

Third, and while "you can achieve scalability to the enterprise only by logical integration," I can't agree with the plain implication of the context of this remark, that a data federation approach is the only appropriate one. Logical integration is also achieved by a federated, object layer approach to integration, and that is the real competitor to Stonebraker's data federation approach, not data warehousing or messaging.

Fourth, the suggestion that application integration should be used "when performing update integration for small numbers of existing applications; while data integration should be used "when you need unified views or are integrating many systems," is a direct consequence of the view and that only logical integration through federated data systems is scalable. But as I've indicated earlier, this view is questionable, and is in no way supported by Stonebraker's brief remarks on the subject.

Fifth, in stating that only data federation can produce the necessary access to both integrated realtime data about rapidly changing conditions, and integrated aged data providing the historical and baseline information needed for survival and competitive advantage, Stonebraker again draws a conclusion that is dependent on his ill-supported critique of application integration and his refusal to consider other artificial systems integration approaches. He believes that only the federated data approach can produce DEI.

But such a conclusion is unsupported by his arguments against application integration and belied by his failure to consider approaches beyond data and application integration. It is also contradicted by the consideration that DEI must be based on more than data integration alone. To remain competitive, businesses must have more than current and accurate data. They must also have well-confirmed models, validated business rules and procedures, and other classes of useful knowledge, including software applications. An artificial system that provides true dynamic integration must handle changes in all components of knowledge and information, and not simply in the data-based ones. It is for this reason that the data federation approach is by its very nature inadequate. It integrates only islands of data, it leaves most of the "islands of information" and knowledge of the enterprise still untouched.

Artificial Information Integration

Artificial Information Integration, like data integration, can be physical or logical. Since information is composed of both data and conceptual commitments, physical integration requires that both be extracted from their original disparate enterprise stores and through a process of object warehousing, very similar to data warehousing, be transformed and migrated to a physical object warehouse. Alternatively, multiple sources of information may be placed in communication through a messaging system in exactly the same way this occurs for data integration.

These physical integration alternatives have the same disadvantages for information integration that Stonebraker identified for data integration. And again, a better solution for the integration problem is provided by federating information sources, just as it was by federating data sources.

An information federation, like a data federation, doesn't migrate data anywhere, it manipulates data in place, according to the business rules specified in the system. It employs multiple distributed application servers along with multiple distributed data stores to maintain a unified view through a common object model. These application servers are called Active Information Managers (AIMs). [5] They provide process control and distribution services to the information federation to synchronize and adjust it to locally determined changes. Finally, like data federations, information federations employ broad ranging connectivity to read from and write to, the distributed data stores and applications of the enterprise.

An information federation, like a data federation, is scalable. It is scalable because: its connectivity to application servers allows it to access applications transparently and also because new information managers can be added as needed to distribute the processing and query load across broadly distributed resources.

Artificial Knowledge Integration

The Artificial Knowledge Integration approach is very much like the information integration approach. The difference is in the nature of the information being processed. In the knowledge federation, knowledge production application servers support formal analytical modeling and data mining, and apply validation criteria to the knowledge production process. Otherwise, the Artificial Knowledge Manager (AKM) [6] uses the same object model, process distribution and control services, and connectivity features used by AIMs. Knowledge federations share the same capability to provide a unified view, ability to transform data in place, and scalability, as data and information federations.

The DKMS Solution to the "Islands of Information" Problem

The application, information, and knowledge integration approaches to enterprise artificial systems integration can all be implemented using Distributed Knowledge Management Architecture (DKMA) [5] [7] [8] [9]. The Key Architectural Components of the DKMS are:

- The Artificial Knowledge Manager (AKM)
- Stateless Application Servers
- Application Servers that maintain State
- Object/Data Stores
- Object Request Brokers (ORBs, e.g., CORBA, DCOM)
- Client Application Components.

I've discussed these in some detail elsewhere [6], and for the most part refer you to that treatment. All the components in the system are integrated by the distributed AKM (or perhaps AIM, if the approach is one of information integration) application server. Here is a summary of its features.

The AKM provides Process Control Services, an Object Model of the DKMS, and connectivity to all enterprise information, data stores, and applications. Figure One illustrates the range of data stores and applications integrated by the AKM. In addition, the AKM provides connectivity to ORBs (not illustrated in the figure).

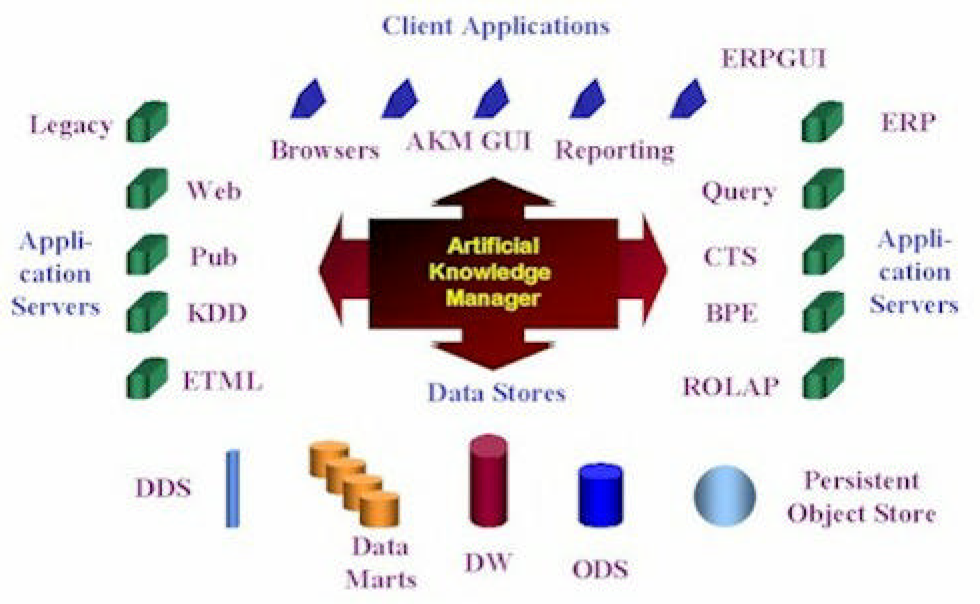

Figure One – An AKM and the Components It Integrates

Sidebar One: Figure One Abbreviations

Web = Web Information Server

Pub = Publication & Delivery Server

KDD = Knowledge Discovery in Databases/Data MiningServers

ETML = Extraction, Transformation, Migration and Loading

DDS = Dynamic Data Staging Area

DW = Data Warehouse

ODS = Operational Data Store

ERP = Enterprise Resource Planning

Query = Query and Reporting Server

CTS = Component Transaction Server

BPE = Business Process Engine

ROLAP = Relational On-Line Analytical Processing

Process Control Services include:

- In -memory proactive object state management and synchronization across distributed objects and

through intelligent agents;
- Component management and Workflow Management through intelligent agents
- Transactional multithreading;
- business rule management and processing; and
- metadata management.

An In-memory Active Object Model/Persistent Object Store is characterized by:
- Event-driven behavior;
- DKMS-wide model with shared representation;
- Declarative business rules;
- Caching along with partial instantiation of objects;
- A Persistent Object Store for the AKM;
- Reflexive Objects.

Connectivity Services have:
- 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; and
- Applications connectivity whether applications are mainframe, server, or desktop - based.

DKM architecture provides a unified view of the enterprise and handles semantic conversions "on-the-fly" through the AKM's object model. It's also scalable to the enterprise level due to (a) its distributed, federated structure, (b) its partial instantiation capability allowing it to load parts of objects into memory (c) its virtual in-memory cached object store, and (d) its broad connectivity to data stores and applications.

The distributed character of the AKM, and its partial instantiation capability are illustrated in Figures Two and Three.

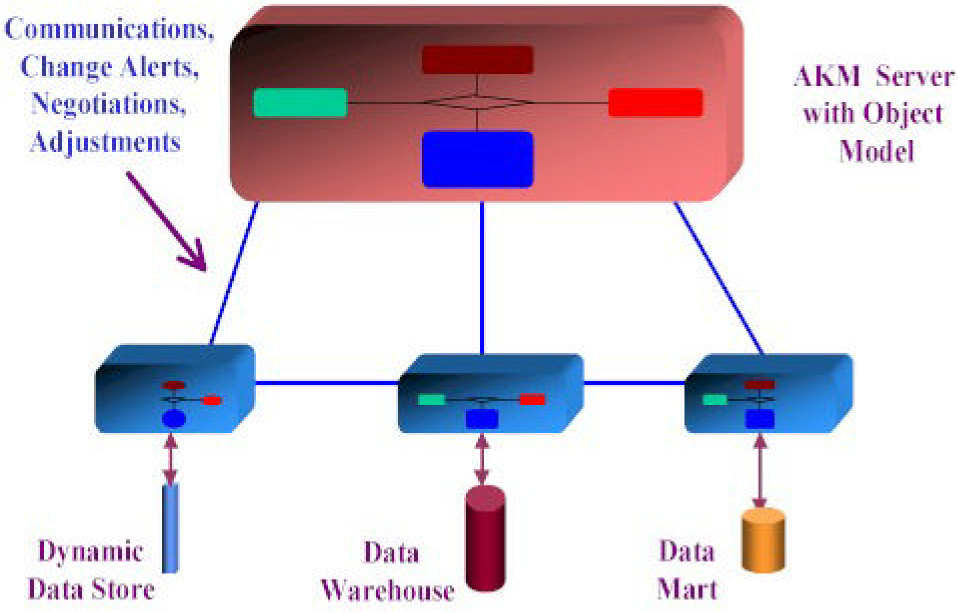
Communications, Change Alerts, Negotiations, Adjustments
AKM Server with Object Model
Dynamic Data Store
Data Warehouse
Data Mart

Figure Two -- A Distributed AKM, Shared Objects and Dynamic Integration

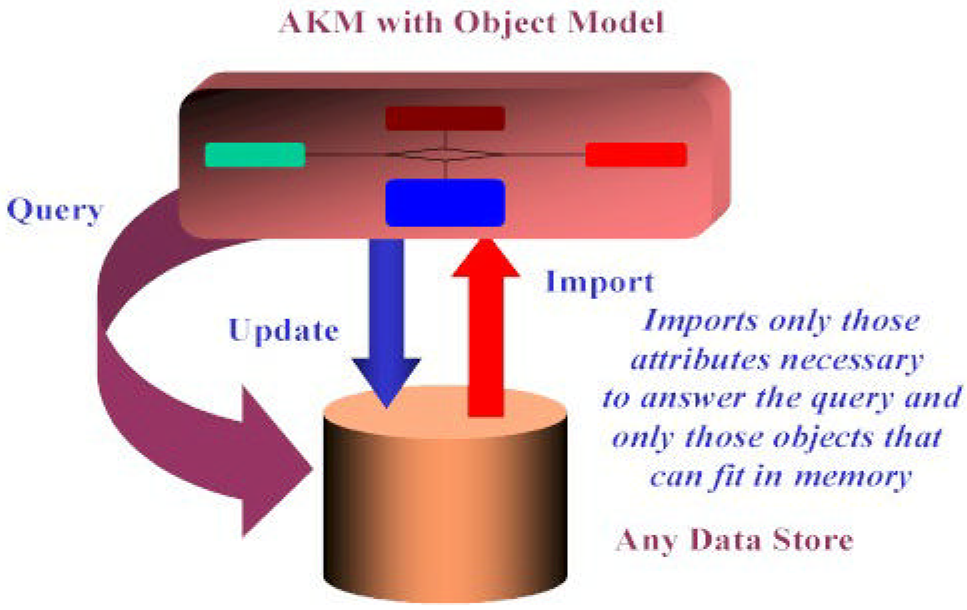
AKM with Object Model
Query
Update
Import
Imports only those attributes necessary to answer the query and only those objects that can fit in memory
Any Data Store

Figure Three -- The AKM and Partial Object Instantiation

The DKMS, The Data Federation and The Enterprise Knowledge Portal

Enterprise Information and Knowledge Portals are subjects of great interest these days. [10][11]. They seem to represent an evolution and convergence of both data warehousing/business intelligence applications and unstructured content management applications. They also represent a very active area of current investment and new business activity.

It is interesting that the success of EIPs and EKPs will probably depend upon the extent to which their back-end is successful in integrating the diverse structured and unstructured data, information, and knowledge accessed through such portals. If integration in the face of change cannot produce data, information, and knowledge consistency, along with acceptable performance, within EIPs and EKPs, these applications will soon lose the favor of business. Viador and Information Advantage, two early entrants into the EIP field from data warehousing, have allied themselves with Stonebraker's Cohera to try to ensure consistency of data across the enterprise and performance across the enterprise. No doubt they realize the vital importance of such consistency to the long-term success of their products.

In other words, the key to the success of EIPs and EKPs is their performance and their capability in adjusting to change in the EIP or EKP application. In the end, the success of EKPs is dependent on architecture, and, in particular, on the success of architectures offering logical integration in providing performance and adaptability.

The issue of which type of federated architecture to use will be a critical issue in determining the success and future of EIP/EKP applications. I believe the approach we follow should not be a data integration approach, or even an information integration approach. I think, instead, it should be a knowledge integration approach.

Those who possess integrated knowledge, after all, have a competitive advantage in decision making over those who possess only integrated data or even integrated information. If this is correct, it means that EKP applications should implement DKM architecture and DKMS solutions rather than data federations.

References

[1] Michael Stonebraker, "United We Stand," *Intelligent Enterprise* (April 20, 1999), 39-45.

[2] Vitria Technology at http://www.vitria.com.

[3] Template Software at http://www.template.com.

[4] New Era Of Networks at http://neonsoft.com.

[5] Joseph M. Firestone,"Architectural Evolution in Data Warehousing," available at http://www.dkms.com/White_Papers.htm

[6] Joseph M. Firestone," The Artificial Knowledge Manager Standard: A Strawman," available at http://www.dkms.com/White_Papers.htm

[7] Joseph M. Firestone, "DKMS Brief No. One: The Corporate Information Factory or The Corporate Knowledge Factory?" at http://www.dkms.com/White_Papers.htm.

[8] Joseph M. Firestone, "DKMS Brief No. Three: Software Agents in Distributed Knowledge Management Systems," at http://www.dkms.com/White_Papers.htm.

[9] Joseph M. Firestone, "DKMS Brief No. Four: Business Process Engines in Distributed Knowledge Management Systems," at http://www.dkms.com/White_Papers.htm.

[10] Christopher C. Shilakes and Julie Tylman, "Enterprise Information Portals," Merrill Lynch, 16 November, 1998

[11] Joseph M. Firestone, "DKMS Brief No. Eight: Enterprise Information Portals and Enterprise Knowledge Portals," at http://www.dkms.com/White_Papers.htm.

Biography

Joseph M. Firestone is an independent Information Technology consultant working in the areas of Decision Support (especially Enterprise Knowledge Portals, Data Warehouses/Data Marts, and Data Mining), Knowledge Management, and Database Marketing. He is developing an integrated Knowledge Discovery in Databases (KDD)/data mining approach incorporating a fair comparison methodology for evaluating data mining results. In addition, he formulated the concept of Distributed Knowledge Management Systems (DKMS) as an organizing framework for software applications supporting Natural Knowledge Management Systems. Dr. Firestone is one of the founding members of the Knowledge Management Consortium, The Chairperson of the KMC’s Artficial Knowledge Management Systems Committee, and a member of its Executive Committee. You can e-mail Joe at eisai@home.com.

[Up] [KMBenefitEstimation.PDF] [MethodologyKIv1n2.pdf] [EKPwtawtdKI11.pdf]
[KMFAMrev1.PDF] [EKPebussol1.PDF] [The EKP Revisited]
[Information on "Approaching Enterprise Information Portals"]
[Benefits of Enterprise Information Portals and Corporate Goals]
[Defining the Enterprise Information Portal]
[Enterprise Integration, Data Federation And The DKMS: A Commentary]
[Enterprise Information Portals and Enterprise Knowledge Portals]
[The Metaprise, The AKMS, and The EKP] [The KBMS and Knowledge Warehouse]
[The AKM Standard]
[Business Process Engines in Distributed Knowledge Management Systems]
[Software Agents in Distributed Knowledge Management Systems]
[Prophecy: META Group and the Future of Knowledge Management]
[Accelerating Innovation and KM Impact]
[Enterprise Knowledge Management Modeling and the DKMS]
[Knowledge Management Metrics Development]
[Basic Concepts of Knowledge Management]
[Distributed Knowledge Management Systems (DKMS): The Next Wave in DSS]