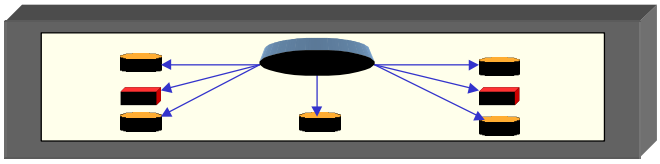
The Artificial Knowledge Manager and Distributed Knowledge Management Systems

By

Joseph M. Firestone, Ph.D.
eisai@moon.jic.com

Object Developers Group(ODG)
February 10, 1999



EIS Web Site: http://www.dkms.com

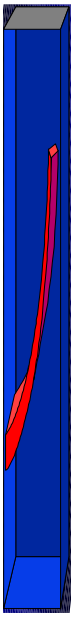
Natural Knowledge Management Systems (NKMS)

- What is an Enterprise NKMS?
- Its interaction properties are not determined by design
- They are emergent
- They determine the outcomes of the enterprise's knowledge processes.
- The NKMS includes organizational components produced by it, such as computers.

Artificial Knowledge Management Systems (AKMS)

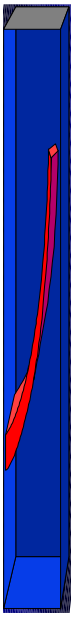
- What is an AKMS?
- The AKMS supports the Knowledge and KM processes of the NKMS
- An AKMS is fully designed; while an NKMS is not
- An AKMS responds predictably to its user inputs; while an NKMS is a self-organizing cas
- Once it starts we can only interact with it and influence it.

AKMS Use Cases



- Two ways to look at the AKMS are in terms of its use cases and its architecture.
- In the Unified Modeling Language (UML) a use case is defined as "a set of sequences of actions a system performs that yield an observable result of value to a particular actor." An actor is a human agent.

AKMS Use Cases (Page Two)



- When an AKMS is viewed functionally as an application, its users perform a set of use cases supporting various tasks within the main activities of the knowledge and KM processes of an NKMS.
- An AKMS doesn't automate all NKMS activities. Only some.
- Figure One shows the abstract relationship of AKMS Use Cases to knowledge and KM processes

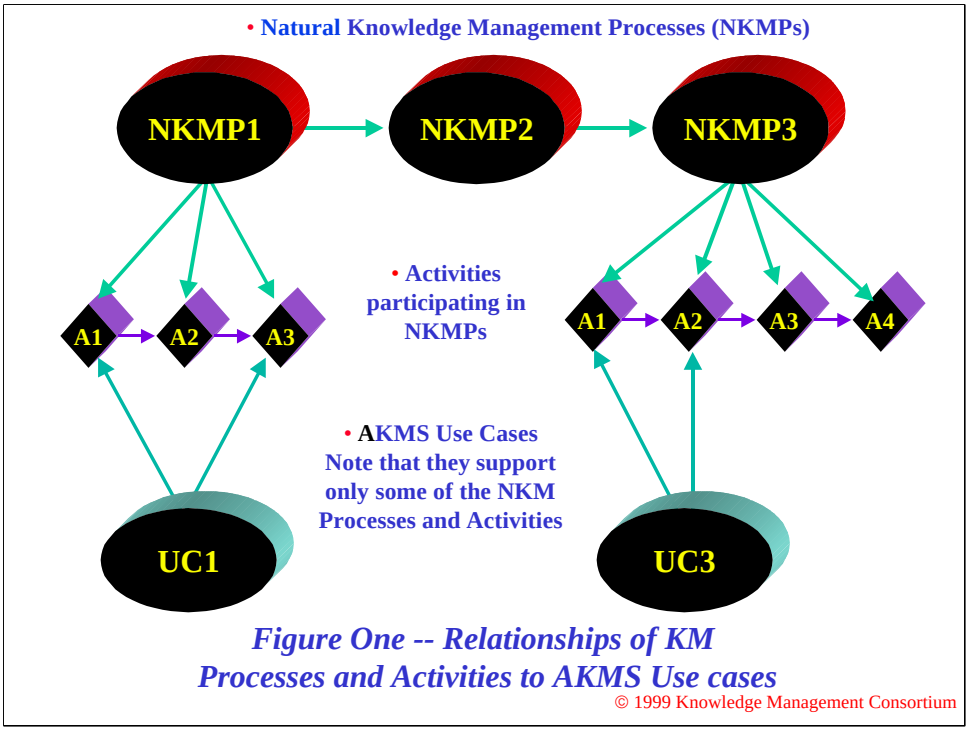
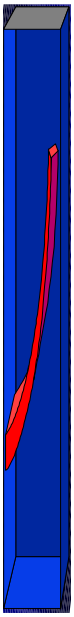


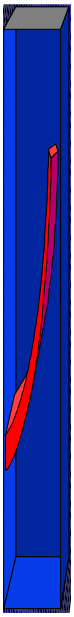
Figure One -- Relationships of KM Processes and Activities to AKMS Use cases

Knowledge and KM Processes



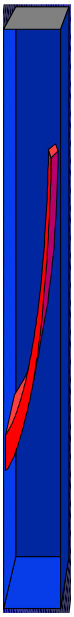
- Knowledge Production
- Knowledge Acquisition
- Knowledge Transmission
- Representing KM
- Leading KM
- KM Knowledge Production
- KM Knowledge Acquisition

Knowledge and KM Processes



- KM Knowledge Transmission
- Changing Knowledge Process Rules
- Handling Crises in Knowledge Processes
- Allocating KM Resources and mandating implementation for Various Knowledge and KM Process activities
- Negotiating KM with business process representatives

AKMS Structure



- If use cases specify the functional or activity aspect of the AKMS, the objects and components of the AKMS that support these use cases, along with their interrelationships provide its structure.
- We can begin to understand AKMS structure by visualizing a basic, abstract architecture.
- That architecture is expressed in Figure Two.

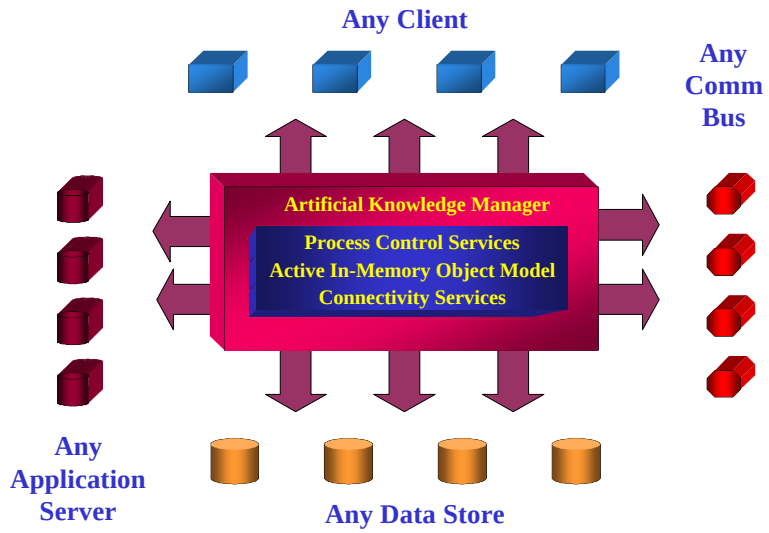


Figure Two -- KMC "Straw Man" AKMS Architecture

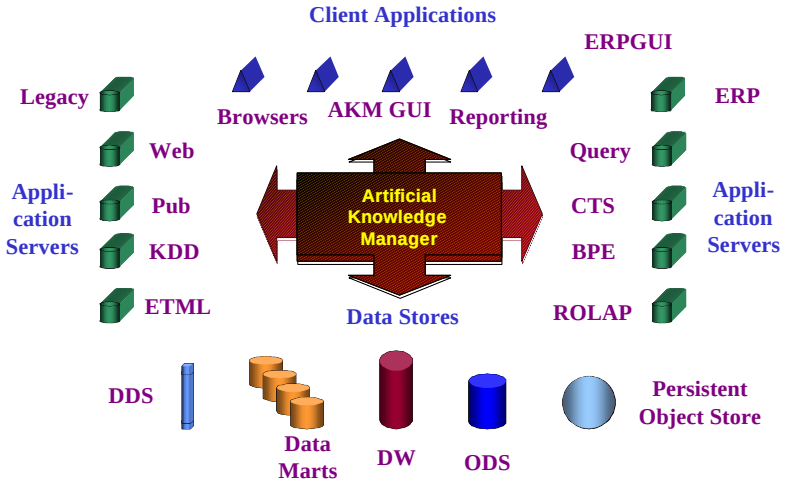
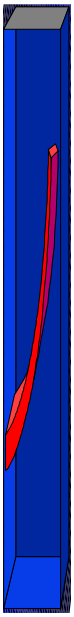


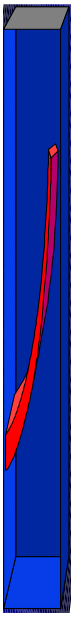
Figure Three -- The AKM, Data Stores, Application Servers, and Clients

The AKMS and the DKMS



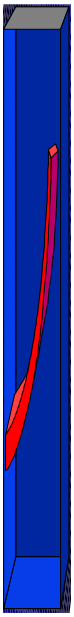
- A Distributed Knowledge Management System (DKMS) of an enterprise is a specific type of AKMS designed to manage the integration of distributed computer hardware, software, and networking objects/ components into a functioning whole supporting enterprise knowledge production, acquisition, and transmission processes.
- The DKMS, in other words supports producing, acquiring and transmitting the enterprise's knowledge base.

The AKMS and the DKMS



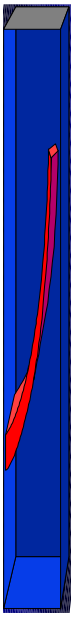
- Current AKMSs are, in fact, DKMSs because distributed systems are the current technological embodiment of the AKMS.
- For the remainder of the talk I will use the terms AKMS and DKMS interchangeably.

AKMS: Diversity and Dynamics



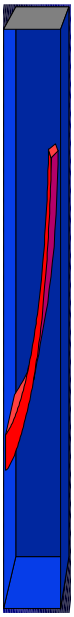
- Figure Three makes clear the diversity of component types in the AKMS.
- It is because of this diversity and its rapid rate of growth in the last few years that the AKM is necessary.
- Change in the AKMS can be introduced through so many sources that if the AKMS is to adapt to change it needs an integrative component like the AKM to play the major role in its integration and adaptation.

AKMS Architecture



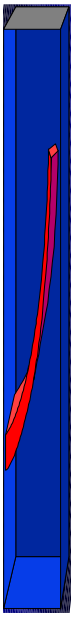
- The Key Architectural Components of the AKMS are:
  - The Artificial Knowledge Manager (AKM)
  - Stateless Application Servers
  - Application Servers that maintain State
  - Object/Data Stores
  - Object Request Brokers (e.g., CORBA, DCOM)
  - Client Application Components
- More detail on these follows.

The AKM



- An AKM provides Process Control Services, an Object Model of the Artificial Knowledge Management System (AKMS) (the system corresponding to the AKMS architecture), and connectivity to all enterprise information, data stores, and applications
  - Process Control Services include:
    - In-memory proactive object state management and synchronization across distributed objects

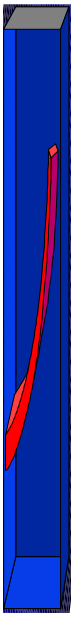
The AKM (Page Two)



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

- In-memory Active Object Model/Persistent Object Store is characterized by:
  - Event-driven behavior
  - AKMS-wide model with shared representation

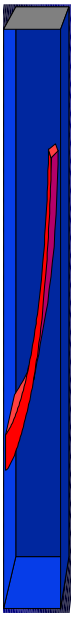
The AKM (Page Three)



- Declarative business rules
- Caching along with partial instantiation of objects
- A Persistent Object Store for the AKM
- Reflexive Objects, and
- Software Agents

- Connectivity Services should have:
  - Language APIs: C, C++, Java, CORBA, DCOM
  - Databases: Relational, ODBC, OODBMS, hierarchical, network, flat file, etc.

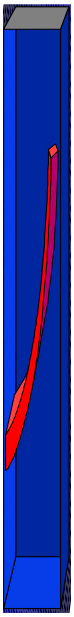
The AKM (Page Four)



- Wrapper connectivity for application software: custom, CORBA, or COM-based.
- Applications connectivity including all the categories mentioned in Figure Three above, whether these are mainframe, server, or desktop - based.

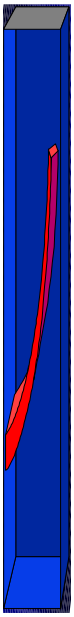
- In the following slides I'll expand on Process Control Services and the Active Object Model.

Process Control Services: Object Management and Synchronization



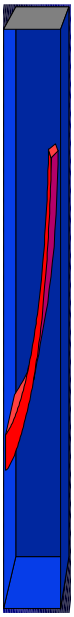
- The AKMS supports a variety of data stores and application servers that allow batch, transaction, and DSS processing to occur in the same system.
- The result of this diversity of processing activities is to introduce frequent and rapid changes into the AKMS, its data stores, and its application servers.

Process Control Services: Object Management and Synchronization (Page Two)



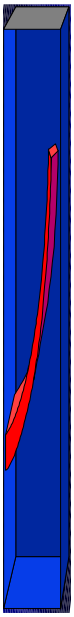
- Change in data, methods (including business rules), and behavior is the “law of life” in the AKMS.
- The problem of managing, synchronizing and adapting to these changes in the AKMS is the Dynamic Integration Problem (DIP).
- A primary function of the AKMS, and its AKM integrative component, is to automate Dynamic Integration (DI) as much as practicable.

Process Control Services: Object Management and Synchronization (Page Three)



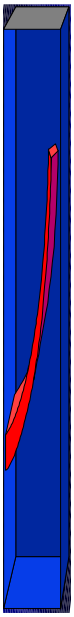
- To perform dynamic integration, the AKM must:
  - look for changes in shared objects and additions to the total pool of objects and relationships,
  - alert all system components sharing the objects of such changes, and also
  - make decisions about which changes should be implemented in each affected component throughout the system.

Process Control Services: Object Management and Synchronization (Page Four)



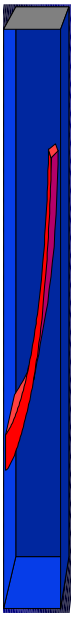
- The AKM accomplishes these tasks by using its in-memory, shared, active object model with its support for event-driven behavior, a common view of the system's objects, declarative business rules, and caching of data along with use of partial instantiation of objects.
- In addition, the AKM relies on a persistent representation of the object model. The objects in the object model are reflexive -- aware of their present state and any change of state.

Process Control Services: Object Management and Synchronization (Page Five)



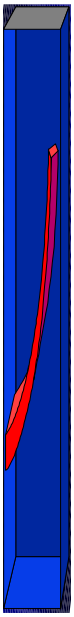
- The AKM accomplishes proactive monitoring and coordinating of changes in its shared objects through their reflexivity and capacity for event-driven behavior, and through software agents. The capacity for event-driven behavior causes the objects to adjust in response to event-induced changes in some shared objects by making corresponding changes in themselves.
- A particular type of AKM event-driven object that is also autonomous is a software agent.

Process Control Services: Component Management and Synchronization



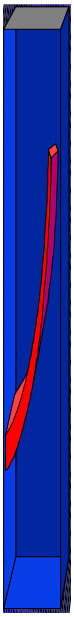
- Component management is the ability to monitor, co-ordinate, and synchronize changes in components, and is analogous to object state management at the component level.
- Component Management and Synchronization in the AKM requires much the same set of capabilities as object state management and synchronization.

Process Control
Services: Work Flow Management



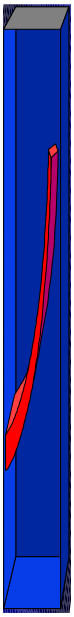
- Work flow refers to the automated system constructed to implement a use case, a part of a use case, or a set of related use cases in a software application.
- The AKM supports management of work flows composed of tasks performed by multiple application servers of diverse processing type.

Process Control
Services: Transactional Multi-threading



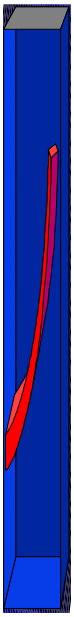
- Transactional multi-threading is the ability to manage each thread within a process as a separate transaction.
- Each thread can represent an instance of an active object.
- This form of multitasking provides the AKM with a powerful form of parallelism useful in work flow management as well as in object and component DI.

Process Control Services: Business Rule and Metadata Management and Processing



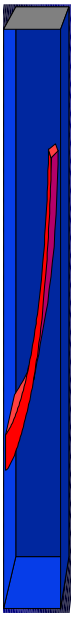
- Business Rule and Metadata Management and Processing are both derivative services of Object and Component Management and Synchronization.
- Business Rules are encapsulated in objects and components as methods, while metadata is encapsulated as attributes.

In-Memory Active Object Model/ Persistent Object Store: Event-Driven Behavior



- The AKM provides an Active Object Model.
- Object methods in the Active Object Model are triggered by (1) events, (2) agents, and /or (3) programmed periodic activation
- Events include user inputs, changes in object attribute values, changes in attributes themselves, or changes in methods themselves.
- Event-driven behavior is implemented in the AKMS through sequences of rules having antecedents and consequents.

In-Memory Active Object Model/Persistent Object Store: AKM with Shared Representation



- Many of the objects in the AKM are shared across distributed physical platforms -- either data stores, or application servers.
- In fact, the AKM may be viewed as a special distributed application server or business process engine that maintains state, shares a set of reflexive objects across physical platforms, and manages and integrates multiple processes changing these shared objects.

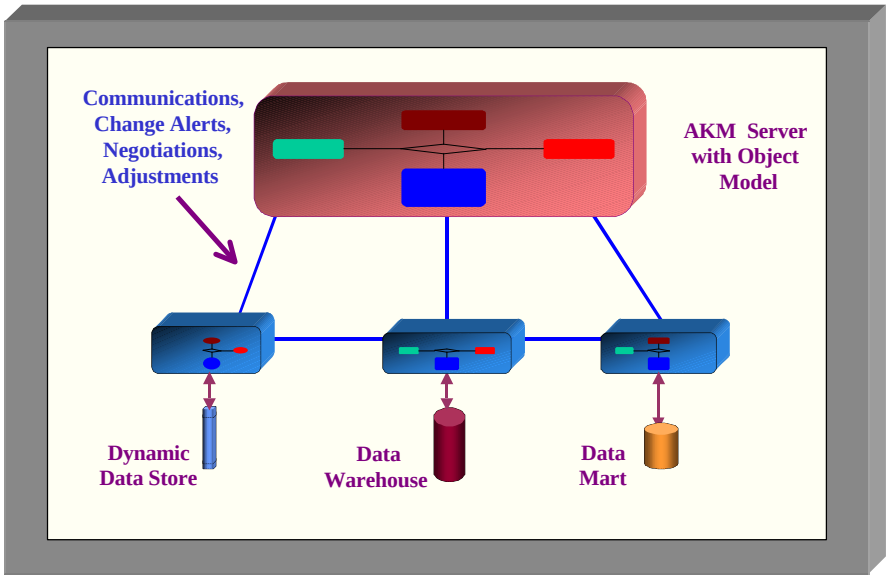
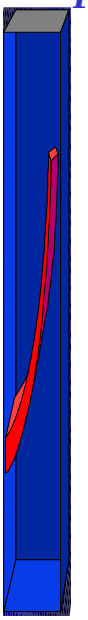


Figure Six -- A Distributed AKM,
Shared Objects and Dynamic Integration

In-Memory Active Object Model/ Persistent Object Store: Declarative Business Rules



- Both declarative and procedural business rule networks are supported as methods in classes and objects of the model.
- Declarative Rule networks are those whose rules fire in parallel to determine an outcome.
- Procedural Rule networks are those whose rules fire in sequence.

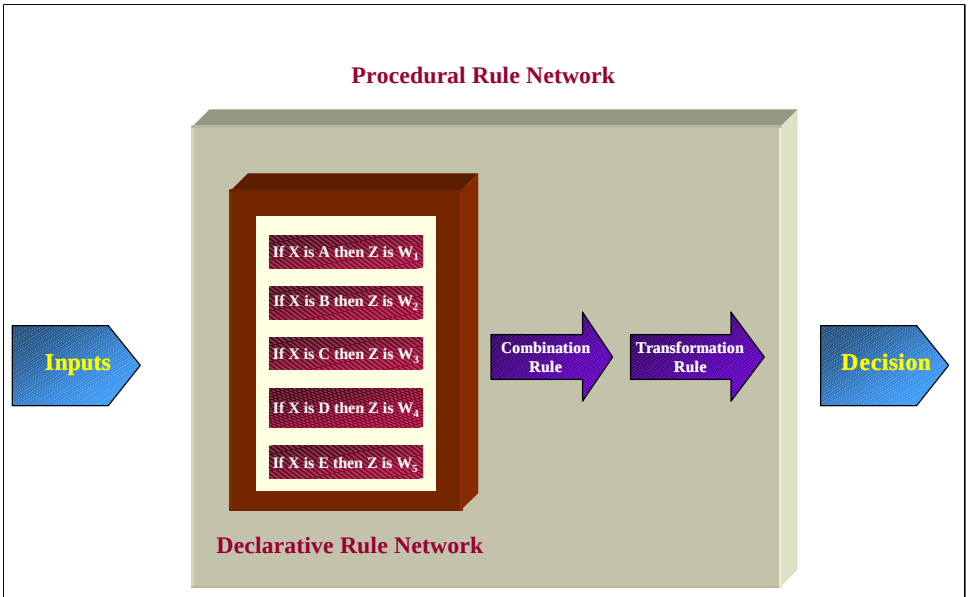
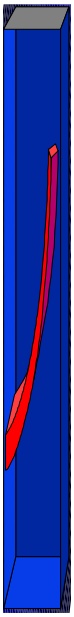


Figure Seven -- Declarative and Procedural Rule Networks

In-Memory Active Object Model/Persistent Object Store: Caching and Partial Instantiation



- The ability to perform partial instantiation of objects is particularly important to the AKM in allowing it to develop rapid query performance.
- In partial instantiation only those attributes called for in a query, and only those records specified are brought into the in-memory object model.
- In this way, the data entering the AKM from data stores in the AKMS can be "chunked," and the amount of data that the AKM must handle can be minimized.

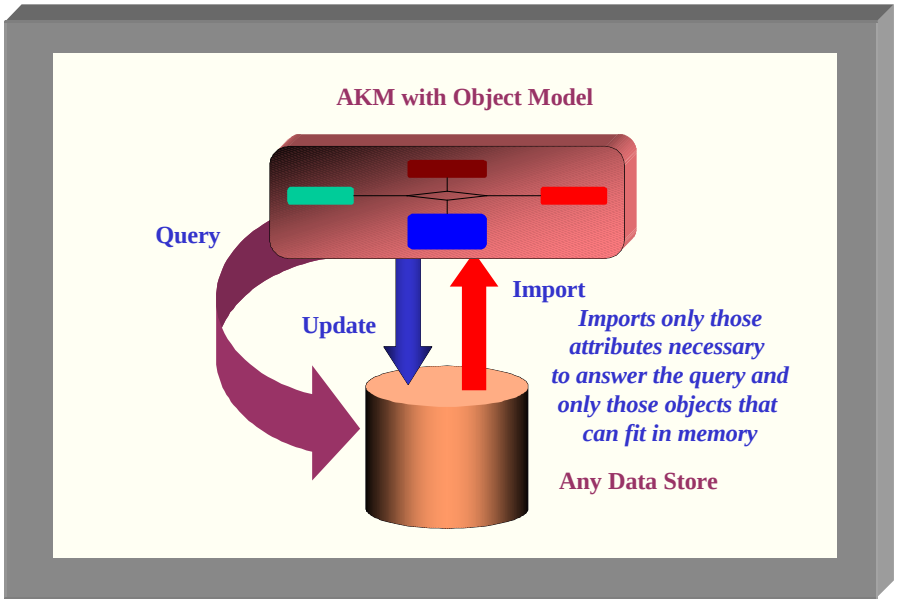
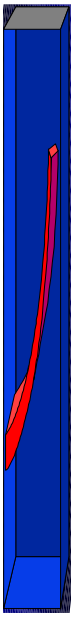


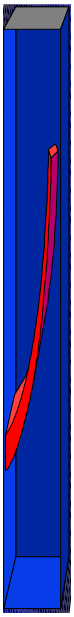
Figure Eight -- The AKM and Partial Instantiation

In-Memory Active Object Model/Persistent Object Store: The Persistent Object Store



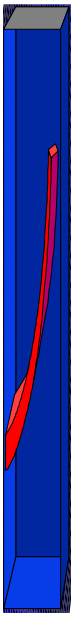
- The AKM uses either a relational database or an OODBMS to store the Active Object Model in persistent form.
- In either case the Active Logical Object Model must be mapped to the physical data model of the database.
- The mapping is straightforward in case of an OODBMS, because the structure of the active object model matches the structure of the database.

In-Memory Active Object Model/ Persistent Object Store: Reflexive Objects



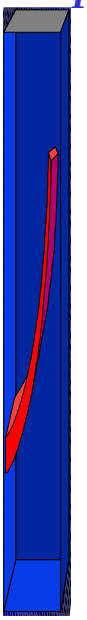
- AKMs use reflexive objects.
- Reflexive Objects are aware of their present state and any change of state.
- When combined with event-driven behavior reflexive objects provide the foundation for automatic propagation of events and changes in state among themselves.

In-Memory Active Object Model/ Persistent Object Store: Software Agents



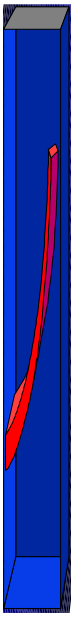
- A Software Agent (SA) is an object that acts on behalf of another object (its client) and behaves to at least some degree:
- autonomously (without continuous direction),
- socially (interacts with other agents),
- proactively (influences its environment), and
- reactively (is influenced by its environment).

In-Memory Active Object Model/ Persistent Object Store: Intelligent Software Agents



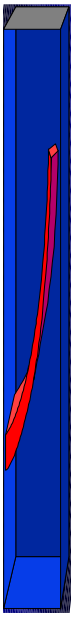
- An intelligent software agent is an SA that:
  - has an in-memory knowledge base including cognitions, evaluations, goals, and perhaps even affects;
  - is rational in the sense that it makes decisions,
  - acts to attain its goals; and
  - learns.

In-Memory Active Object Model/ Persistent Object Store: Static and Mobile Agents



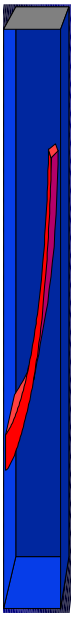
- A static SA is one that does not move from the platform that creates it.
- A Mobile SA can move across a network from one physical computer to another.
- It can do this autonomously, as it perceives the need for such movement.
- The "source computer" of a mobile SA is its home agency.
- The AKM is a home agency for mobile agents.

In-Memory Active Object Model/ Persistent Object Store: Agents and the AKM



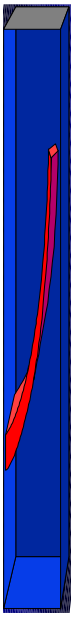
- Mobile SAs and their agencies require a host environment in order to execute.
- This is a distributed computing environment overlaying a host distributed computing environment.
- It provides various essential services to mobile SAs, including the ability to create them, and the ability to execute.
- This environment is the AKM. And, in a larger sense, the AKMS.

Application Servers



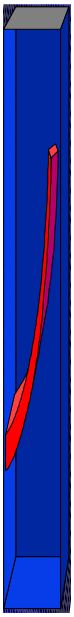
- The development of multi-tier distributed processing systems was characterized by the appearance of application servers.
- Application servers provide services to other components in a distributed processing system by executing business logic and data logic on data accessed from database servers.

Stateless Application Servers



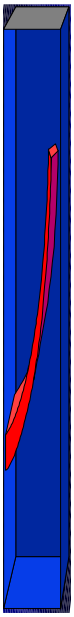
- "Business state is the information that describes the momentary status of the organization."
- “To create business state, most applications acquire data from a database and then load it into memory for manipulations by the user."
- This is the "stateless" approach because, in it, a back-end database, rather than internal memory, manages state.

BPEs: Application Servers That Maintain State



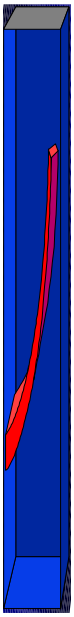
- "Business Process Engines manage the most important business state both in a fast in-memory environment and in close coordination with back-end databases."
- Because of their in-memory maintenance of state BPEs process many user requests without help from a database.
- In addition, they specialize in complex business rule processing, because their ability to maintain state is a special advantage in performing such processing.

Types of BPEs



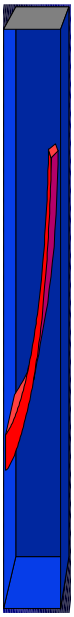
- An important aspect of specifying the AKMS is specifying the current universe of application servers and projecting the appearance of new types.
- Here are some criteria for defining types of Business Process Engines:
  - whether they are distributed across physical components or not;
  - whether a BPE application server deals with a single or multiple business processes;
  - the business process the BPE supports.

Types of BPEs (Page Two)



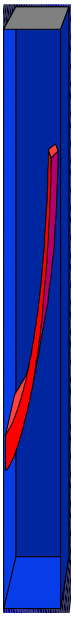
- An AKM is just a BPE that is both distributed and encompasses all of an AKMS's processes.
- A multi-process BPE can fall short of being an AKM, and instead can be restricted to a cluster of related processes.
- So, there are at least three types suggested by this criterion: a single process BPE, a BPE cluster, and an AKM.

Knowledge and KM Process BPEs



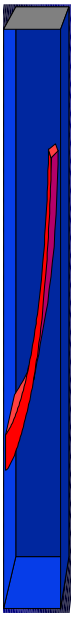
- The third criterion for classifying BPEs is the business process supported. Here is an incomplete classification of BPE application servers based on knowledge, KM and Data Warehousing sub-processes.
  - Collaborative Planning;
  - Extraction, Transformation, and Loading (ETL);
  - Knowledge Discovery in Databases (KDD);
  - Knowledge base/object/component model maintenance, and change management (The AKM);

Knowledge and KM Process BPEs (Page Two)



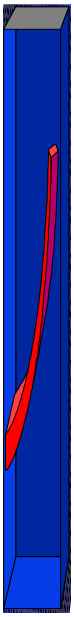
- Knowledge Publication and Delivery (KPD);
- Computer-Based Training (CBT);
- Report Production and Delivery (RPD);
- ROLAP Application Server;
- Operational Data Store (ODS);
- Forecasting/Simulation Server;
- ERP servers,
- Financial Risk Management,
- Telecommunications Service Provisioning,
- Transportation Scheduling,
- Stock Trading Servers, and
- Work Flow servers.

Object/Data Stores



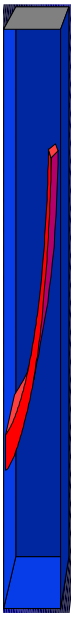
- There are few, if any, limits on the object/data stores in the AKMS.
- Legacy data, flat files, Relational Data Bases, Object Relational Data Bases, OODBMSs, multidimensional data stores, and vertical technology databases all fit within the AKMS.
- The AKMS must also integrate Image, Text, Report, Video, Audio, and File Document Types.

Object Request Brokers



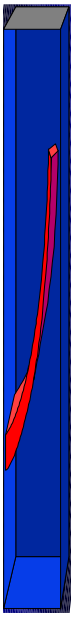
- ORBs
  - provide an intermediate layer between clients and servers in a distributed network
  - receive requests from clients and selects servers to satisfy the requests
  - can activate appropriate servers
  - can translate data between clients and servers
- The AKM must support CORBA and DCOM ORBs to fulfill its integrative function.

An AKM Standard



- The Artificial Knowledge Management System Committee (AKMSC) of the Knowledge Management Consortium (KMC) is working on an AKM Standard.
- The first meeting of the AKMSC, held in Silver Spring, MD on January 29, agreed on the following statement of the scope and deliverables of the standards effort

AKMSC Scope and Deliverables



- Scope of the AKM Committee:
  - Context: The AKMS supports Knowledge Processes and Knowledge Management Processes ranging from the individual to the enterprise levels of interaction in the enterprise context.
  - Standards: AKMSC leverages active and developing international standards in specifying AKMS elements.

AKMSC Scope and Deliverables: Steps

- Steps in Specifying AKM System (Threaded KMC Forum on Each) follow.
- Specify AKMS Use Case Model and Relate to EKM Processes and Activities
- Specify the Artificial Knowledge Manager (AKM) Logical Component
- Specify Types of Client Application Components.

AKMSC Scope and Deliverables: Steps (Page Two)

- Specify Types of Application Servers
- Specify Communication Buses including Object Request Brokers (ORBs)
- Specify Types of Data Stores
- Specify AKMS Architectural Model
- Specify AKMS Model
- Specify Artificial Knowledge Manager Standard

AKMSC Scope and Deliverables: Steps (Page Three)

- Deliverables: Proposed standards for an Artificial Knowledge Management System (AKMS), and Artificial Knowledge Manager (AKM), expressed in a Draft Technical Report.