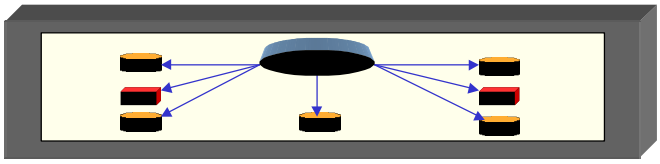
Executive Information Systems, Inc.

Project Four: Standard On The Artificial Knowledge Manager

By

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

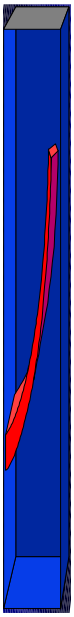
Prepared for First KMC/AIIM
KM ANSI/ISO Standards Committee Meeting
January 29, 1999



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

15 seconds

AKMS Use Cases

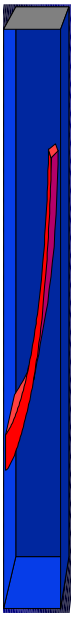


- Two ways to look at the AKMS are in terms of its use cases and its architecture.
- In the Unified Modeling Language (UML) a use case is defined as "a set of sequences of actions a system performs that yield an observable result of value to a particular actor." An actor is a human agent.

30 seconds, 45 seconds

- Yesterday, I talked about the distinction between the NKMS and the AKMS, provided an overview of the AKMS, and outlined a process to arrive at an AKM standard.
- Today, we will go into the AKMS in more detail.

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

30 seconds, 75 seconds

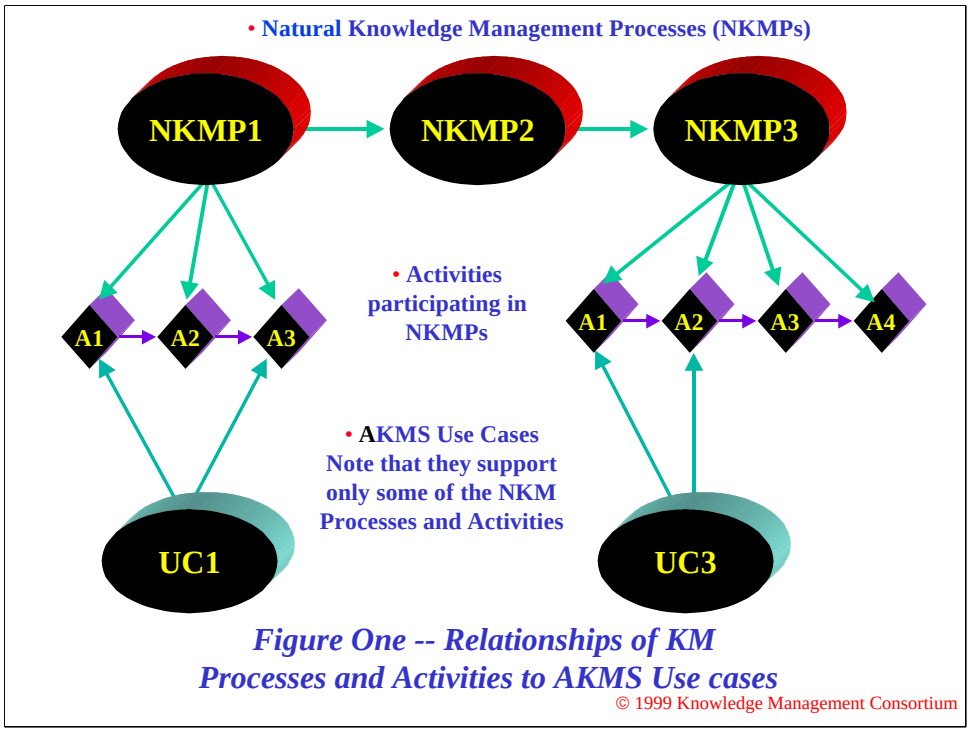
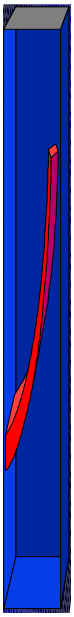


Figure One -- Relationships of KM Processes and Activities to AKMS Use cases

30 seconds, 105 seconds

- I showed Figure One yesterday. So, I won't go through it again, but just display to refresh memories.
- The main point is that UCs can be mapped to activities and NKMPs, but they don't fully automate the AKMPs.

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 most abstractly in Figure Two.

30 seconds, 130 seconds

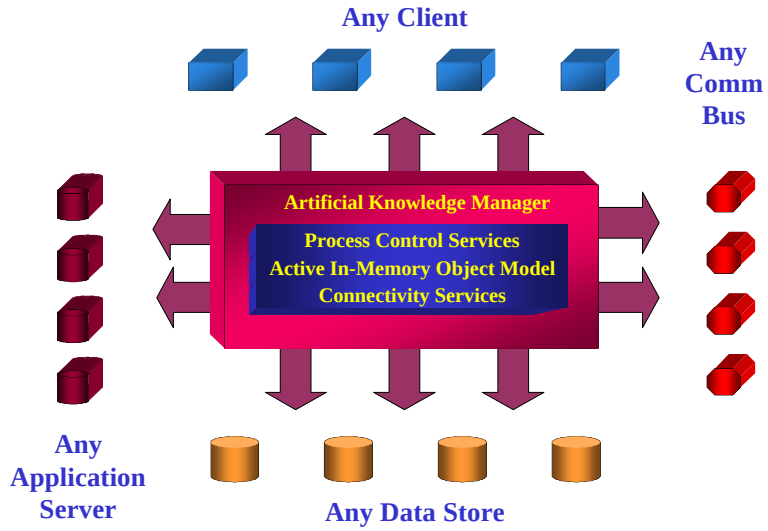


Figure Two -- KMC "Straw Man" AKMS Architecture

- 60 seconds, 190 seconds, 3 minutes 10 seconds.
- The figure, again first shown yesterday, shows clients, application servers, communication buses and data stores integrated through a single logical component called an Artificial Knowledge Manager (AKM).
- I will specify the AKM in much more detail later. For now a more concrete visual picture showing the variety of component types in the AKMS is given in Figure Three.

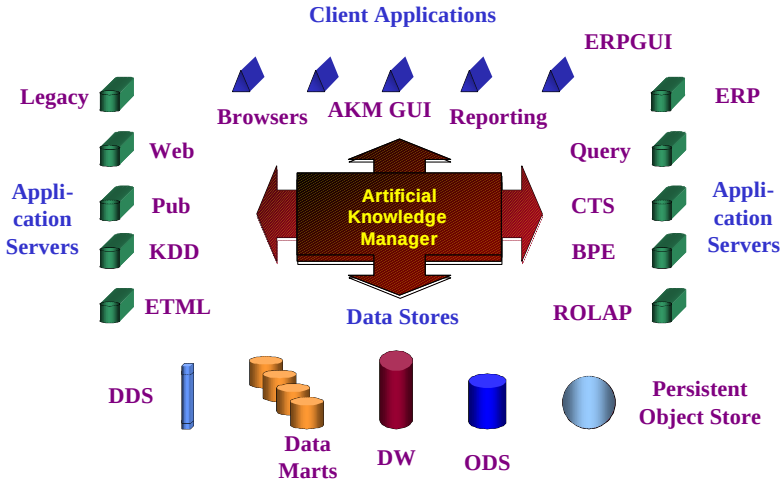
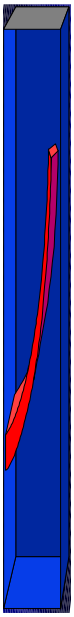


Figure Three -- The AKM, Data Stores, Application Servers, and Clients

- 90 seconds, 280 seconds, 4 minutes, 40 seconds
- An important difference between the two figures is that the communications bus aspect of the AKMS is implicit in figure Three, where I've assumed that the AKM incorporates it.
- I’ve covered the detail in the Figure Yesterday.
- Any questions, on the detail? Please hold substantive questions on the architecture for later.

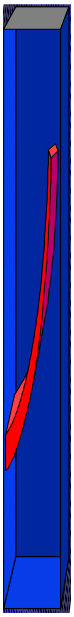
AKMS: Diversity and Dynamics



- 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.

30 seconds, 310 seconds, 5 minutes, 10 seconds

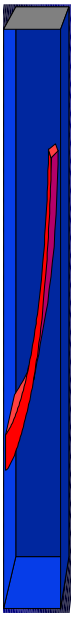
AKMS Architecture



- 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.

30 seconds, 340 seconds

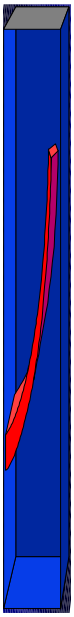
The AKM



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

30 seconds, 370 seconds

The AKM (Page Two)

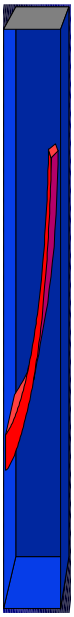


- Component management and Workflow Management through intelligent agents
- Transactional multithreading
- business rule management and processing, and
- metadata management.

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

30 seconds, 400 seconds

The AKM (Page Three)

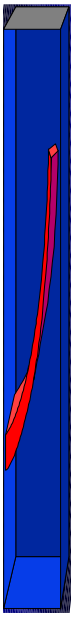


- Declarative business rules
- Caching along with partial instantiation of objects
- A Persistent Object Store for the AKM
- Reflexive Objects, and
- Software Agents

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

30 seconds, 430 seconds, 7 minutes, 10 seconds

The AKM (Page Four)

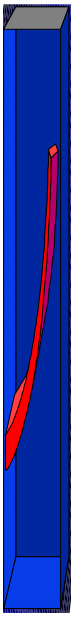


- Wrapper connectivity for application software: custom, CORBA, or COM-based.
- Applications connectivity including all the categories mentioned in Figure Three above, whether these are mainframe, server, or desktop - based.

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

30 seconds, 460 seconds

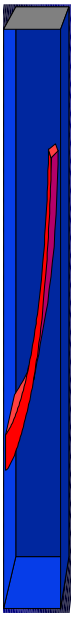
Process Control Services: Object Management and Synchronization



- 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.

30 seconds, 490 seconds

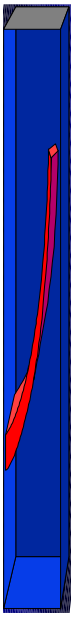
Process Control Services: Object Management and Synchronization (Page Two)



- 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.

30 seconds, 520 seconds

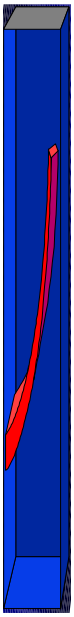
Process Control Services: Object Management and Synchronization (Page Three)



- 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.

30 seconds, 550 seconds, 9 minutes, 10 seconds

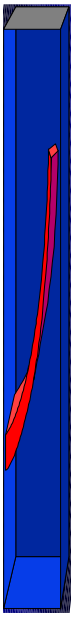
Process Control Services: Object Management and Synchronization (Page Four)



- 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.

30 seconds, 580 seconds = 9 minutes, 40 secs.

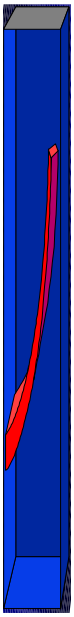
Process Control Services: Object Management and Synchronization (Page Five)



- 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.

30 seconds, 610 seconds

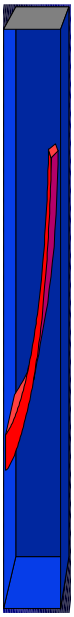
Process Control Services: Component Management and Synchronization



- 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.

- 50 seconds, 660 seconds, 11 minutes
- Like objects, components can also be shared across applications and physical platforms. And they also change frequently and rapidly and require DI.
- 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. It too, needs to be performed in real-time, and it too requires proactive, in-memory operation to be most effective.
- Component Management and Synchronization in the AKM requires much the same set of capabilities as object state management and synchronization.

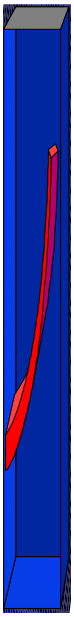
Process Control
Services: Work Flow Management



- 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.

- 45 seconds, 705 seconds
- 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.
- For example, a collaborative planning work flow application involving a planning business process engine and multiple database servers can be integrated by an AKM.
- Another example is an integrated database marketing workflow involving ETML, Operational Data Store, DSS Database, Data Mining, Business Process Engine, and Web Server components.

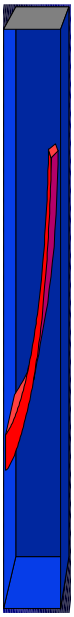
Process Control Services: Transactional Multi-threading



- 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.

- 45 seconds, 750 seconds
- 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.
- Because they support transactional multi-threading, AKMs provide for multiple objects, belonging to different classes, to reside in the same process.
- This form of multitasking allows for concurrent execution of disparate business rules associated with different objects
- It provides the AKM with a powerful form of parallelism useful in work flow management as well as in object and component DI.

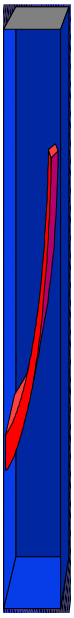
Process Control Services: Business Rule and Metadata Management and Processing



- 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.

- 30 seconds, 780 seconds, 13 minutes
- 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.
- So part of what we mean when we refer to object and component state management and synchronization is management and processing of business rules and metadata.

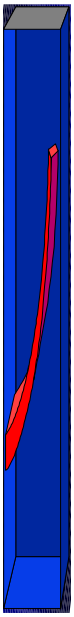
In-Memory Active Object Model/ Persistent Object Store: Event-Driven Behavior



- 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.

- 60 seconds, 840 seconds, 14 mins.
- The AKM provides an Active Object Model. It is distributed. Much of it is shared across physical platforms. And it can be either persistent or resident in-memory
- 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.
- Events can trigger agent behavior which then follows an autonomous course in implementing adjustments.
- 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.

- 30 seconds, 870 seconds
- 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.
- It is this sharing of objects and components across platforms that creates a common view of the AKM and its metadata.
- Figure Six illustrates the role of Shared Objects in the AKM and in DI.

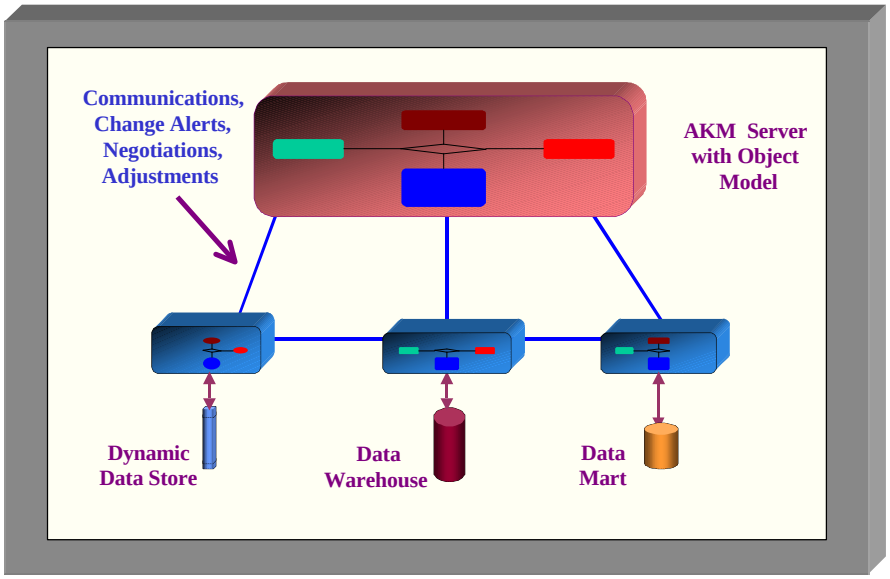
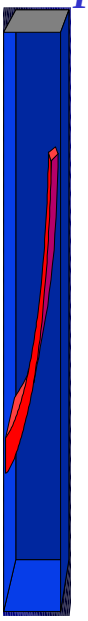


Figure Six -- A Distributed AKM, Shared Objects and Dynamic Integration

30 seconds, 900 seconds, 15 minutes

- At the top is an AKM component with its object model.
- In the middle are other AKM components with their versions of the object model.
- On the bottom are data stores associated with the distributed AKM Servers through a mapping of the object model to the physical model of the data store.
- Dynamic Integration is achieved through communications alerts, negotiations, and adjustments among the AKM components of the distributed AKM Server, and between the server and the data stores it is associated with.

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.

- 45 seconds, 945 seconds
- 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.
- Event-driven behavior in the AKM is frequently determined by sequences of declarative rules or rule networks constituting procedural rule networks.
- Agent-driven behavior is triggered by events but then is determined by the agent's autonomous program.

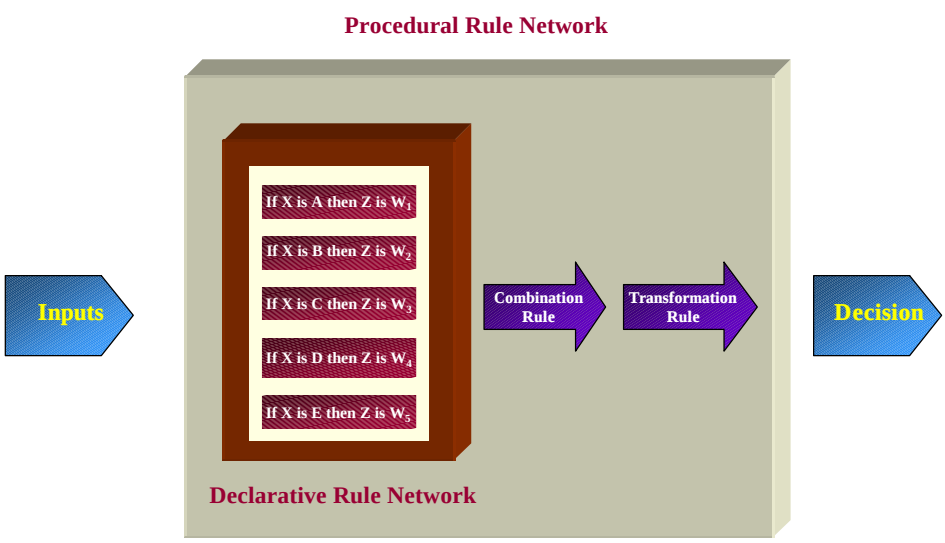
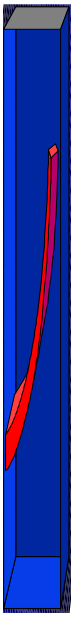


Figure Seven -- Declarative and Procedural Rule Networks

30 seconds, 975 seconds, 16 minutes 15 secs.

- The Procedural Rule Network consists of the declarative rule network, the combination rule network, and the transformation rule.

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.

- 45 seconds, 1020 seconds, 17 mins.
- 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.
- As a result, it is much more likely that the difficult processing involved in any query can be done in the AKM's "virtual database" in-memory. **Figure Eight** illustrates partial instantiation of objects by an AKM.

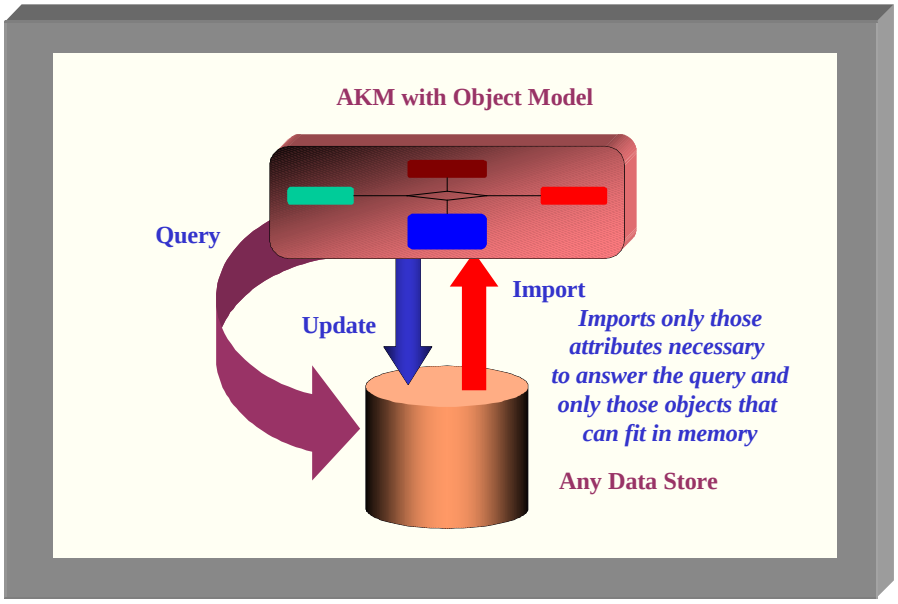
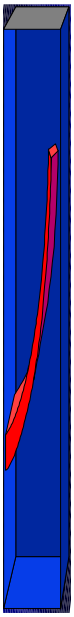


Figure Eight -- The AKM and Partial Instantiation

30 seconds, 1050 seconds

- Here's an AKM mapped to a data store.
- It can update the data store.
- It can query it.
- It imports results back, but it grabs data only for those attributes necessary to answer the query,
- and it grabs in chunks no larger than available AKM memory resources.

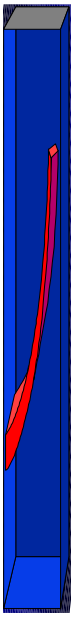
In-Memory Active Object Model/Persistent Object Store: The Persistent Object Store



- 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.

- 30 seconds, 1080 seconds, 18 mins.
- 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. There is no “impedance mismatch,” because there is no need to unwrap the logical objects and map their attributes onto physical table columns.
- This is not the case with an RDBMS, and if one is used for persistent storage of the object model a performance penalty is paid.

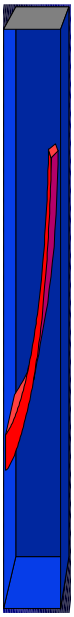
In-Memory Active Object Model/ Persistent Object Store: Reflexive Objects



- 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.

- 30 seconds, 1110 seconds
- AKMs use reflexive objects.
- Reflexive Objects are aware of their present state and any change of state.
- In this way they are like agents in Natural Knowledge Management and other business processes.
- When combined with event-driven behavior reflexive objects provide the foundation for automatic propagation of events and changes in state among themselves.

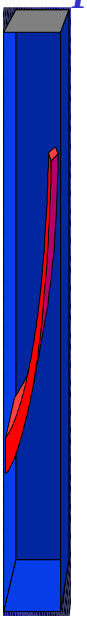
In-Memory Active Object Model/ Persistent Object Store: Software Agents



- 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).

- 30 seconds, 1140 seconds, 19 minutes
- 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).

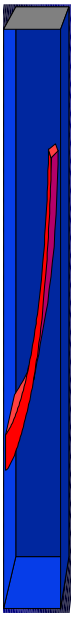
In-Memory Active Object Model/ Persistent Object Store: Intelligent Software Agents



- 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.

- 30 seconds 1170 seconds

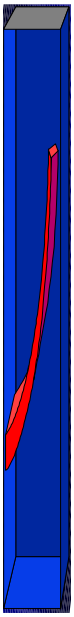
In-Memory Active Object Model/ Persistent Object Store: Static and Mobile Agents



- 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.

- 45 seconds, 1215 seconds, 20 minutes, 15 secs.
- 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 agency consists of a computing environment, an agent scripting capability, and a database.
- The AKM is a home agency for mobile agents.

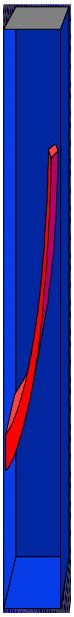
In-Memory Active Object Model/ Persistent Object Store: Agents and the AKM



- 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.

- 30 seconds, 1245 seconds

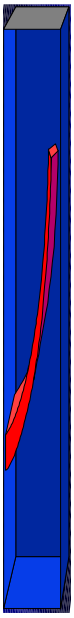
Application Servers



- 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.

- 30 seconds, 1275 seconds, 21 minutes 15 seconds.
- 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.
- The class of application servers is sub-divided by Rymer's distinction between "stateless" and in-memory server environments. Application Servers with Active in-memory Object Models he calls Business Process Engines (BPEs), a name similar to Vaskevitch's Business Process Automation Engines.

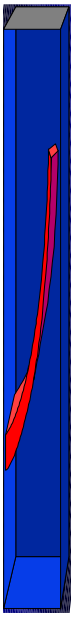
Stateless Application Servers



- "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.

- 45 seconds, 1350 seconds, 22 minutes, 30 seconds
- According to Rymer: "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.
- Among stateless application servers Rymer distinguishes:
  - Web Information Servers (they provide access to databases from web browsers);
  - Component Servers (they "provide data access and interaction frameworks for software components"); and
  - Transaction Processing Monitors (they coordinate transactions within a distributed system).

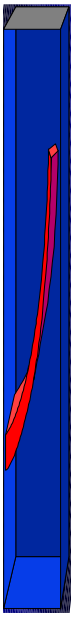
BPEs: Application Servers That Maintain State



- "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.

- 30 seconds, 1380 seconds, 23 mins.
- "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.
- KM software applications such as KDD/data mining servers, publication and delivery servers, the AKM itself, and many other server types are all BPEs..
- The job of the AKMS is to integrate the burgeoning list of BPEs into an enterprise wide system.

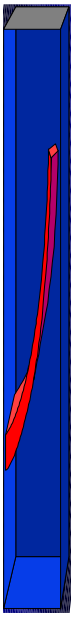
Types of BPEs



- 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.

- 45 seconds, 1425 seconds. 23 mins., 45 secs.
- Therefore, 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.
- Distributed BPEs can be a powerful tool for upgrading performance in AKMSs, as well as for integrating their various components.

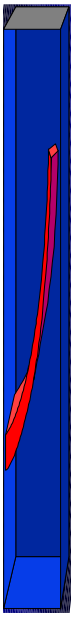
Types of BPEs (Page Two)



- 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.

- 60 seconds, 1485 seconds. 24 minutes, 45 secs.
- 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.
- How well a multi-process BPE performs will be correlated to the extent of its distribution, and to the complexity of the process it must support.
- But holding complexity constant, single process, non-distributed BPEs will generally perform better than multi-process non-distributed BPEs.
- So, multi-process BPEs will generally be distributed BPEs

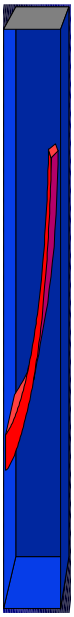
Knowledge and KM Process BPEs



- 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);

30 seconds, 1515 seconds, 25 minutes, 15 seconds

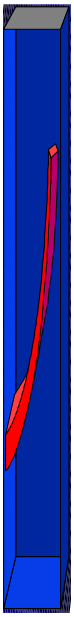
Knowledge and KM Process BPEs (Page Two)



- 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.

30 seconds, 1545 seconds, 25 minutes, 45 seconds

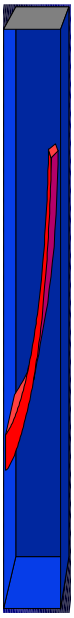
Object/Data Stores



- 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.

- 30 seconds, 1575 seconds. 26 mins., 15 seconds
- There are few, if any, limits on the object/data stores in the AKMS.
- These data stores incorporate objects, components, or their attributes in a non-volatile persistent form.
- Legacy data, flat files, Relational Data Bases, Object Relational Data Bases, OODBMSs, multidimensional data stores, and vertical technology databases all fit within the AKMS.
- In addition, the AKMS must also integrate Image, Text, Report, Video, Audio, and File Document Types.
- That is, it is the job of the AKMS to develop and maintain connectivity to various data stores, and not simply DBMSs.

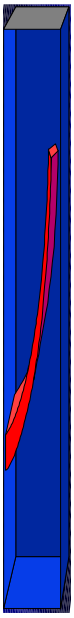
Object Request Brokers



- 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.

- 30 seconds, 1605 seconds, 26 minutes, 45 seconds
- ORBs provide an intermediate layer between clients and servers in a distributed network.
- The ORB receives requests from clients and selects servers to satisfy the requests.
- The ORB can activate appropriate servers.
- The ORB can translate data between clients and servers
- Generally, ORB servers are stateless and therefore are not BPEs (though this is not part of ORB specifications).
- The AKM must support CORBA and DCOM ORBs to fulfill its integrative function. That is, it must be able to act as both CORBA and DCOM Servers and Clients.
- In this way the AKM, with its greater integrative functionality, can be built “on top of” an ORB standard.

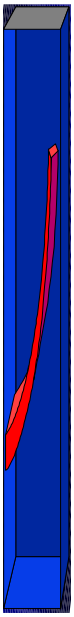
The Unified Knowledge Language



- The KMC is currently developing a standard on the Unified Knowledge Language (UKL).
- This standard will specify a contextually rich language that can represent and transmit knowledge from one software program or device to another.
- The specification will contain a message structure for the transmission of knowledge via the AKM.

- 30 seconds, 1635 seconds, 27 minutes, 15 secs.
- The KMC is currently developing a standard on the Unified Knowledge Language (UKL).
- This standard will specify a contextually rich language that can represent and transmit knowledge from one software program or device to another. The language will consist of syntax, rules, and format.
- The specification will contain a message structure for the transmission of knowledge via the AKM.
- So, as time goes on, the AKMSC will need to coordinate with the UKL Committee in refining our standard.

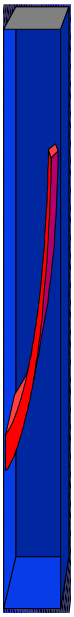
Knowledge and KM Processes



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

- 45 seconds, 1680 seconds, 28 minutes
- Here is a list of knowledge and KM processes to help in approaching the task of specifying use cases
- I've included a list of use cases matched to these processes and associated activities in Table One of Working Paper No. One.
- We can use this list as a basis for developing a use case model for the AKMS.

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

30 seconds, 1710, 28 minutes, 30 seconds

- Figures Two and Three above, suggest a basic "straw man" program for specifying the AKMS and the AKM Standard. It is as follows:

AKMSC "Strawman" Program

- 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.
- Specify Types of Application Servers
- Specify Communication Buses including Object Request Brokers (ORBs)

30 seconds SubT = 1740 seconds, 29 minutes

- Specify AKMS Use Case Model and Relate to EKM Processes and Activities: 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." A use case model requires description of the all the use cases and associated diagrams
- Specify the Artificial Knowledge Manager (AKM) Logical Component: We'll talk about this component in a minute.
- Specify Types of Client Application Components: These refer to Interface Components of Applications
- Specify Types 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.
- Specify Communication Buses including Object Request Brokers (ORBs): Which ORBS are essential for an open standard?

AKMSC "Strawman" Program

- Specify Types of Data Stores
- Specify AKMS Architectural Model
- Specify AKMS Model
- Specify Artificial Knowledge Manager Standard
- Specify Knowledge Warehouse Standard

30 seconds SubT = 1770 seconds = 29 minutes 30 seconds

- Specify Types of Data Stores: RDBMS, OODBMS, Flat File, etc.
- Specify AKMS Architectural Model: An abstract of the AKMSs technical structure
- Specify AKMS Model: An abstract of component structure, relationships and dynamics in the AKMS
- Specify Artificial Knowledge Manager Standard: Define requirements for the AKM logical component
- Specify Knowledge Warehouse Standard: Define the knowledge warehouse in the context of the AKMS and the AKM.
- Clearly, there's an appreciable amount of work associated with these specification tasks. We can begin work now ;
- First considering whether to retain, modify, or reinvent the straw man program.
- Then we can divide into subcommittees according to our interests and specialization in order to do the work over the longer term.