

Just-in-Time Knowledge Management

Larry Kerschberg and Hanjo Jeong

E-Center for E-Business, Department of Information and Software Engineering,
George Mason University, MSN 4A4, Fairfax, Virginia, 22030-4444
{kersch, hjeong}@gmu.edu, <http://eceb.gmu.edu/>

Abstract. This paper presents the requirements for just in time knowledge management (JIT-KM). In order to deliver high-value information to user for decision-making, one must understand the user's preferences, biases and decision context. A JIT-KM architecture is presented consisting of user, middleware and data services to search for information from heterogeneous sources, and to rank and deliver this to decision-makers. The search process is described using concepts from Knowledge Sifter, an agent-based system that accesses heterogeneous sources using Semantic Web Services.

1 Introduction

The concept of just-in-time knowledge management (JIT-KM) is appealing in that the goal is to provide the *right information*, to the *right people*, at the *right time* – just in time – so they can take action based on that information. While the just-in-time concept originated with Toyota in its drive to improve its manufacturing processes, the concept can also be applied to the timely delivery of information. There are a number of inter-related current trends that impact our study of JIT-KM. They are: On Demand Computing, On Demand Business, On Demand Retail, and On Demand Organizations.

On Demand Computing allows users to treat the computing infrastructure as an *information utility*, which can marshal the required resources (computers, storage, etc.) and charge based on usage. Users do not have to be aware of where the computers and storage facilities reside – they are virtual. On Demand Organizations, or Virtual Organizations, can be configured on the fly from existing Web services offered by vendors. The goal is dynamically configure a collection of Web services by searching for candidates, negotiating with service providers for quality-of-service agreements, vetting the selected services, composing them, orchestrating their workflow and managing the virtual organization life-cycle [8, 9].

Both On Demand Computing and On Demand Organizations are based on the *virtualization* of resources and services that are then managed on behalf of users to deliver the desired functions. They are both related to the notions of GRID computing [6] and Semantic Web Services [20, 26]. The amount of meta-data [13, 27] required to manage these virtual environments is considerable.

On Demand Business integrates the enterprise with its suppliers by optimizing business processes and the supply chain to reduce inventories. On Demand Retail

treats stores shelves as space to be managed by suppliers who are paid when customers actually purchase the merchandise. The main concept is the *integration and interoperation of information* among business partners and suppliers to achieve a high degree of transparency and efficiency among their business processes. In a recent *New York Times* article [7], Wal-Mart, the world's largest retailer, was using its 460-terabyte data warehouse to monitor worldwide operations in near real-time. They have created an extranet called Retail Link that allows suppliers to see how well their products are selling. Eventually, Wal-Mart will have the capability to conduct *scan-based trading* in which the supplier will own the product until it is scanned for purchase. This will reduce inventory costs for Wal-Mart. Wal-Mart will be requiring its major suppliers to use RFID tags on its shipments, in order to keep track of inventory as it enters the warehouses.

This example indicates that Wal-Mart is providing virtual shelf space for its suppliers; they are responsible for stocking those shelves and maintaining their inventories in a just-in-time fashion. Wal-Mart collects and uses massive amounts of data to obtain an up-to-the-minute picture of world-wide operations and can make command decisions based on their analysis of the data.

The above trends inform the concept of Just-in-Time Knowledge Management, which might also be called On Demand Knowledge Management. The sheer volume of data and information available to us make it imperative that we be able to sift and winnow through the mountains of data to find those *knowledge nuggets* so crucial to effective decision-making.

This paper is organized as follows. Section 2 deals with issues associated with requirements and technologies for JIT-KM. Section 3 presents Knowledge Sifter, which is an agent-based search tool based on Semantic Web Services. We stress that search is crucial to JIT-KM. The section also presents a meta-model for Knowledge Sifter that can be used to populate a repository with user queries, associated results and user feedback. Section 4 presents a service-oriented architecture for JIT-KM and discusses some of the services needed to support JIT-KM. Section 5 presents our conclusions.

2 Just-in-Time Knowledge Management

The notion of Just-in-Time Knowledge Management (JIT-KM) is that the right information should be available to decision-makers at the right time and in the right place, just in time. This simple concept has very widespread implications for the systems needed to support it. First, how do we determine what is the right information? How do we know who should receive that information? What is the right format of the information based on the decision-maker's location, context, and type of presentation device? How can we capture and represent the user's preferences, bias, context, and most importantly, his or her information requirements?

We explore these issues within the context of a research project called Knowledge Sifter, which is being conducted at George Mason University's E-Center for E-Business. We show how the Knowledge Sifter architecture can be used for JIT-KM.

2.1 Requirements for JIT-Knowledge Management

In order to deliver JIT-information, we must ensure that the information is timely, authentic, trusted and tailored to the decision-maker's needs. These *knowledge nuggets* are pieces of information that make a quantifiable difference in the decision-making process. Timeliness is important in that out-of-date (stale) information can be irrelevant to the current context and task. Authenticity and trust are related in that the decision-maker should have confidence in the information and the sources that provided it. Finally, there are the issues of *data lineage* and *data provenance*, that is, how was the data processed to derive the information? What is the quality of the original and derived data, and how reliable is the source of the data?

These issues are all crucial for the decision-makers to have confidence in the information products, how they were derived, and the assessment of the quality and reliability of the data provider.

2.2 Technologies for JIT-Knowledge Management

The JIT-information requirements suggest that active technologies are needed to support the timely delivery of JIT services. These include *pull scenarios* in which standing queries are posed to heterogeneous sources, both internal to the enterprise and external such as the Internet and World Wide Web. These queries represent items of interest to the decision-maker and evidence substantiating an existing decision scenario would help him to take action. Alternatively, *push scenarios* deliver content to users via syndication such as RSS feeds or by means of specialized subscriptions services. These messages can be searched for content related to the decision-maker's profile and alerts generated and delivered to a preferred device.

An important component of JIT-KM is the use of *meta-data* — data about data — to model and manage the JIT services. Metadata is important in capturing data lineage as information objects are processed throughout the various phases of the activity life cycle. This may include the evolution of user preferences; historical information regarding the results of standing- and ad-hoc queries; the ranking of search results; the authoritativeness of data sources; and the user's perceptions regarding the quality and timeliness of information provided. We now address these issues in the context of the Knowledge Sifter research.

3 The Knowledge Sifter Architecture

The Knowledge Sifter project, underway at George Mason University, has as its primary goals: 1) to allow users to perform ontology-guided semantic searches for relevant information, both in-house and open-source, 2) to refine searches based on user feedback, and 3) to access heterogeneous data sources via agent-based knowledge services. Increasingly, users seek information outside of their own communities to open sources such as the Web, XML-databases, and the emerging Semantic Web.

The Knowledge Sifter project also wishes to use open standards for both ontology construction and for searching heterogeneous data sources. For this reason we have chosen to implement our specifications and data interchange using the Web Ontology Language (OWL) [4, 30], and Web Services [2] for communication among agents and information sources.

3.1 Knowledge Sifter Agent-Based Web Services Framework

The rationale for using agents to implement intelligent search and retrieval systems is that agents can be viewed as autonomous and proactive. Each agent is endowed with certain responsibilities and communicates using an Agent Communication Language [5]. Recently, Huhns [10] has noted that agents can be thought of a Web services, and this is the approach we have taken to implement the agent community comprising Knowledge Sifter. The family of agents presented here is a subset of those incorporated into the large vision for Knowledge Sifter. This work is motivated by earlier research into Knowledge Rovers [11, 12] performed at GMU. This research is also informed by our previous work on WebSifter, [15, 16, 18] a meta-search engine that gathers information from traditional search engines, and ranked the results based on user-specified preferences and a multi-faceted ranking criterion involving static, semantic, categorical and popularity measures.

The Knowledge Sifter architecture [13, 14] may be considered a service-oriented architecture consisting of a collection of cooperating agents. The application domain we are considering is that of Image Analysis. The Knowledge Sifter conceptual architecture is depicted in Figure 1. The architecture has three layers: User Layer, Knowledge Management Layer and Data Layer. Specialized agents reside at the various layers and perform well-defined functions. This collection of cooperating agents supports interactive query specification and refinement, query decomposition, query processing, ranking, as well as result ranking and presentation. The Knowledge Sifter architecture is general and modular so that new ontologies and new information resources can be easily incorporated [22]. The various agents and services are described below.

User and Preferences Agents

The User Agent interacts with the user to elicit user preferences that are managed by the Preferences Agent. These preferences include the relative importance attributed to terms used to pose queries, the perceived authoritativeness of Web search engine results, and other preferences to be used by the Ranking Agent. The Preferences Agent can also learn the user's preference based on experience and feedback related to previous queries.

Ontology Agent

The Ontology Agent accesses an imagery domain model, which is specified in the Web Ontology Language (OWL). In addition, there are two authoritative name services: Princeton University's WordNet and the US Geological Survey's GNIS.

They allow the Ontology Agent to use terms provided by the name services to suggest query enhancements such as generalization or specialization.

For example, WordNet can provide a collection of synonyms for a term, while GNIS translates a physical place in the US into latitude and longitude coordinates that are required by a data source such as TerraServer. Other appropriate name and translation services can be added in a modular fashion, and the domain model would be updated to accommodate new concepts and relationships. We now discuss the various sources used by the Ontology Agent.

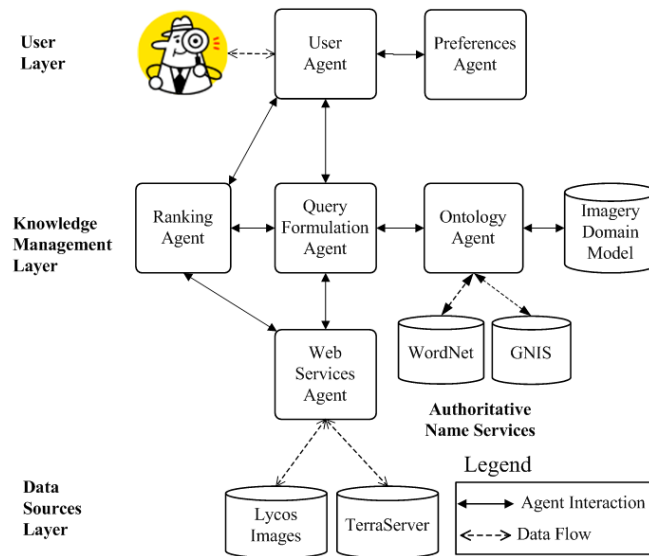


Fig. 1. The Knowledge Sifter Agent-Based Web Services Architecture

Imagery Domain Model and Schema

The principal ontology used by Knowledge Sifter is the Imagery Domain Model, specified using the Web Ontology Language, OWL. A UML diagram of the ontology is provided in Figure 2.

The class *Image* is defined as having *source*, *content*, and file descriptive *features*. Subcategories of content are *person*, *thing*, and *place*. Since we are primarily interested in satellite and geographic images, the class *place* has two general attributes, *name* and *theme*, together with the subclasses *region* and *address*. The *Region* is meant to uniquely identify the portion of the Earth's surface where the place is located, either by a *rectangle* or a *circle*. In the case of a rectangle we need two latitude values (*north* and *south*) and two longitude values (*east* and *west*), while to specify a circle we need the *latitude* and *longitude* of its center point, and a *radius*. The *address* of our location is identified by *country*, *state*, *city*, *zip code* and *street*. Each image belongs to a specific online source, the *server*, and has *URI-1* as a unique identifier, together with a secondary *URI-2* for a thumbnail (if any). Some qualitative and quantitative attributes are also modeled as subclasses of the general class *features*,

namely *resolution* (in square meters per pixel), *projection* and *datum* (for future GIS utilizations), a *date* range, and *image size* (with *height* and *width* expressed in pixels).

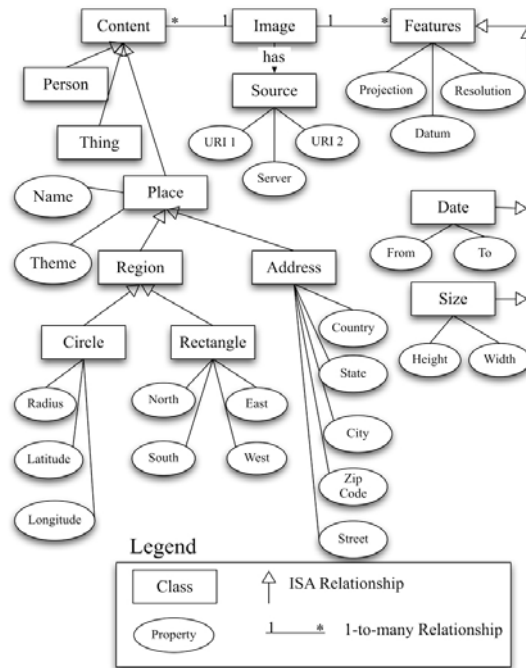


Fig. 2. Ontology Schema in the Unified Modeling Language Notation

Authoritative Name Services

The two name services are WordNet, developed at Princeton University, which is a lexical database for the English language. When the initial query instance, specifying whether a person, place, or thing, is sent to the Ontology Agent, it then consults WordNet to retrieve synonyms. The synonyms are provided to the Query Formulation Agent to request that the user select one or more synonyms. The decision is communicated to the Ontology Agent which updates the appropriate attribute in the instantiated version of the OWL schema. If the attribute value is the name of a class of type *place* then the Ontology Agent passes the instance to the USGS GNIS.

The second name service is the USGS Geographic Names Information System (GNIS) which is a database of geographic names within the United States and its territories [29]. GNIS was developed by the USGS and the U.S. Board on Geographic Names to meet major national needs regarding geographic names and their standardization and dissemination. It is an integration of three separate databases, the National Geographic Names Data Base, the USGS Topographic Map Names Data Base, and the Reference Data Base. Records within the database contain feature name, state, county, geographic coordinates, USGS Geographic Map name, and

others. Other specialized name and translation services can be integrated into Knowledge Sifter and linked to the Domain Model.

Query Formulation Agent

The user indicates an initial query to the Query Formulation Agent. This agent, in turn, consults the Ontology Agent to refine or generalize the query based on the semantic mediation provided by the available ontology services. Once a query has been specified by means of interactions among the User Agent and the Ontology Agent, the Query Formulation Agent decomposes the query into subqueries targeted for the appropriate data sources. This involves semantic mediation of terminology used in the domain model ontology and name services with those used by the local sources. Also, query translation is needed to retrieve data from the intended heterogeneous sources.

For example, if the user specifies the domain of his search as *place*, Lycos and TerraServer will be chosen. In cases of *person* and *thing*, only Lycos will be chosen. In the case of person and thing, the user is asked to choose a specific meaning from the list retrieved from WordNet, and then the synonym set and hypernym set regarding that particular meaning are retrieved. Synonyms can be chosen as alternate names. Hypernyms can be used to generalize the user's concept. The terms chosen by the user are used to query Lycos. For example, if the user specifies the concept 'Rushmore' the following synonym set is returned by WordNet:

Rushmore, Mount Rushmore, Mt. Rushmore – (a mountain in the Black Hills of South Dakota; the likenesses of Washington and Jefferson and Lincoln and Roosevelt are carved on it)

In this case, the synonym set {Rushmore, Mount Rushmore, Mt. Rushmore} and the hypernym set {Mountain Peak} are retrieved from WordNet. If user chooses "Mount Rushmore" and "Mountain Peak", two different queries, "Mount AND Rushmore" and "Mountain AND Peak" are posed to Lycos, because the Lycos image search doesn't support the logical connector "OR" in search terms.

In the case of place, the user-selected synonym set and hypernym set are requested from GNIS server using a similar approach, that is, the queries ("Mount AND Rushmore" and "Mountain AND Peak") are posed to the GNIS server in order to collect a list of locations from which the user can choose. The user can specify a state to restrict the GNIS results. After the user chooses one specific location, the name of the location is also used to submit queries to the Lycos server. Concurrently, a query is sent to TerraServer Web service with the appropriate latitude and longitude for the selected place.

In our future research, we will endow the Query Formulation Agent with more rules and policies to help it to make more intelligent decisions about query specification and query optimization. For example, in the case of image databases, a strategy might be to query the image metadata, retrieve and view thumbnails, and then request the collection of selected images. In addition, Knowledge Sifter will have a repository of processed queries, instantiated and annotated according to the OWL schema. This information will be used by the Query Formulation Agent as a Case Base that can be searched and the results reused. For example, a user query might be

specified in stages, and the Case Base could be used to retrieve a relevant query processing strategy, send a request to the Web Services Agent and the results returned for user consideration. If needed, the Ontology Agent could assist in query enhancement as described above.

Web Services Agent

The main role of the Web Services Agent is to accept a user query that has been refined by consulting the Ontology Agent, and decomposed by the Query Formulation Agent. The Web Service Agent is responsible for the choreography and dispatch of subqueries to appropriate data sources, taking into consideration such facets as: user preference of sites; site authoritativeness and reputation; service-level agreements; size estimates of subquery responses; and quality-of-service measures of network traffic and dynamic site workload [21].

The Web Services Agent transforms the subqueries to XML Protocol (SOAP) requests to the respective local databases and open Web sources (TerraServer or Lycos) that have Web Service published interfaces; this is the case for the TerraServer, while Lycos provides an HTTP interface.

Ranking Agent

The Ranking Agent is responsible for compiling the sub-query results from the various sources, ranking them according to user preferences, as supplied by the Preferences Agent, for such attributes as: 1) the authoritativeness of a source which is indicated by a weight – a number between 0 and 10 – assigned to that source, or 2) the weight associated with a term comprising a query.

Data Sources and Web Services

At present, Knowledge Sifter consults two data sources: Lycos Images and the TerraServer. The Lycos server supports keyword-based image search via the web page <http://multimedia.lycos.com>. It makes use of both an image server and external data sources such as web pages for the image search. For a Lycos image search, no advanced search is supported and only conjunctions of terms are used. Therefore, the user cannot specify the image metadata such as *size* or *resolution*, so the results of search are limited. To address these problems the Query Formulation Agent generates a collection of conjunctive and disjunctive queries, while the evaluation and ranking process is left to the Ranking Agent.

The TerraServer is a technology demonstration for Microsoft. There is a Web Service API for TerraServer. TerraServer is an online database of digital aerial photographs (DOQs – Digital Orthophoto Quadrangles) and topographic maps (DRGs – Digital Raster Graphics). Both data products are supplied by the U.S. Geological Survey (USGS). The images are supplied as small tiles and these can be made into a larger image by creating a mosaic of tiles. The demonstrator at terraserver-usa.com uses a mosaic of 2x3 tiles.

Our purpose is to take the ontology-enhanced query and generate specific subqueries for the TerraServer metadata. The resulting image identifiers and their metadata are wrapped into an instance of our image ontology. And an array of these is returned to the Web Service Agent to compile with other results.

3.2 Knowledge Sifter End-to-End Scenario

Consider the following scenario in which a user wishes to search for the term ‘Rushmore’. This scenario shows how the various agents, name services, and data sources interact in handling a user query.

1. The user provides the User Agent with a keyword query: ‘Rushmore’.
2. The user identifies the term as being a person, place or thing via radio buttons in the query form. The user has chosen ‘Place’.
3. The User Agent passes the query to Query Formulation Agent.
4. The Query Formulation Agent invokes the Ontology Agent to instantiate an OWL schema for the ‘Place’ with Name = ‘Rushmore’.
5. The Ontology Agent chooses a service agent based on the initial query. In this case, it requests from WordNet a list of concepts for ‘Rushmore’. WordNet then passes the results back to the Ontology Agent which then passes the results to the User Agent via the Query Formulation Agent for the user decision.
6. The user chooses the ‘Mount Rushmore’ concept, which has three synonyms (‘Rushmore’, ‘Mt. Rushmore’, and ‘Mount Rushmore’).
7. The Ontology Agent then submits the synonym set to the USGS Geographic Name Information Server and receives a list of candidate geographic coordinates.
8. The list of candidate coordinates is sent to the Query Formulation Agent and the user chooses the desired location.
9. The Ontology Agent then updates the OWL schema instance with the chosen latitude and longitude.
10. The Query Formulation Agent then passes the fully-specified query to the Web Service Agent.
11. The Web Services Agent forwards appropriate sub-queries to both Lycos and TerraServer. The TerraServer and Lycos data sources are queried, and the results are sent back to the Web Services Agent. The results are compiled into new OWL instances that describe image metadata.
12. All results are combined and sent to the Query Formulation Agent.
13. The Query Formulation Agent sends the result sets and the original query to the Ranking Agent for ranking.
14. Within the Ranking Agent the image metadata for each returned item is ranked using the weights and preferences provided by the Preferences Agent. The Preferences Agent maintains the user preferences.
15. The Ranking Agent generates a score for each image result, and returns the scored list to the User Agent.
16. The User Agent then sorts the results by ranking and presents them to the user.
17. The user can then select an item from the list to download and view the image.

3.3 Knowledge Sifter Meta-Model

The previous sections have described how the cooperative agents and web services support the search for relevant knowledge from both local and open-source data sources. The end-to-end scenario shows how the various agents and sources interact. The OWL schema is instantiated with information regarding query and its various

transformations into the final ranked results. In this section we elaborate on this concept by presenting a meta-model of the Knowledge Sifter framework so that relevant information can be captured regarding the *lineage* and *provenance* of all aspects of the search process, from query specification, to query reformulation, web service decomposition, results ranking and recommendation presentation. This includes information on the various Knowledge Sifter activities (managed by agents), the outcomes of those activities, the quality of the ranked results, measures of Web service performance, and the authoritative and reliability of data sources.

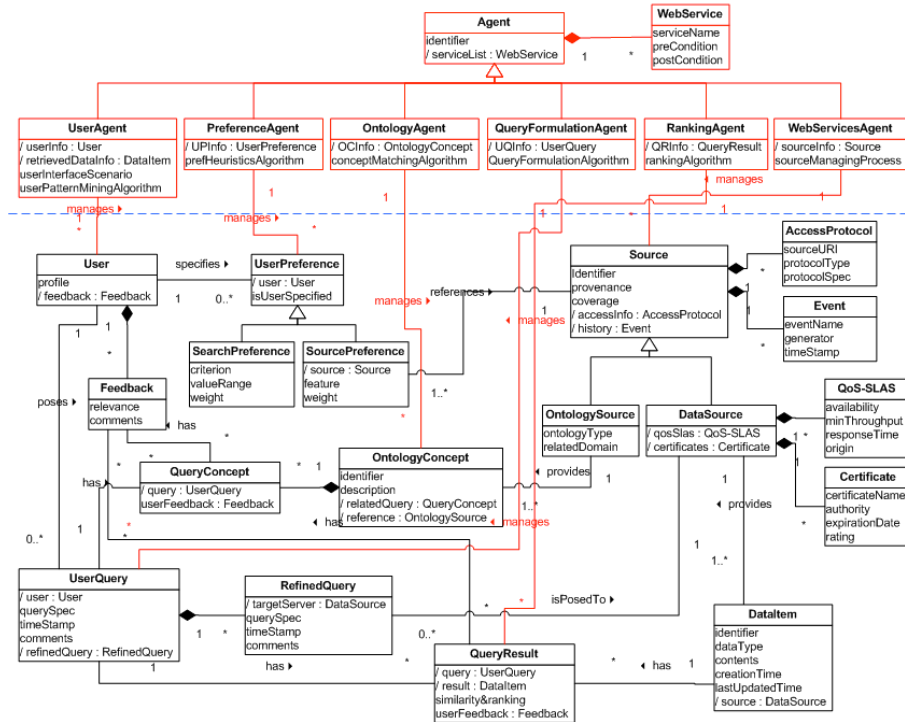


Fig. 3. The Knowledge Sifter Meta-Schema

This meta-model is then used to capture and store metadata for future analysis, filtering and mining for JIT-KM. By stepping back and abstracting the agents, classes, their relationships and properties, we can construct the Knowledge Sifter Meta-Model (KSMM) [25]. Figure 3 depicts the UML Static Model for the KSMM. At the top is the Class Agent, which is specialized to those agents in the KS framework, specifically the UserAgent, PreferencesAgent, OntologyAgent, QueryFormulationAgent, RankingAgent and WebServicesAgent. These agents manage their respective object classes, process specifications, and WebServices. For example, the UserAgent manages the User Class, the UserInterfaceScenario, the User PatternMiningAlgorithm, and the WebServices. The User specifies User Preferences that can be specialized to Search Preferences and Source Preferences. The User poses UserQuery that has several QueryConcept, which in turn relates to an OntologyConcept. The Ontology Agent manages both the UserQuery and the

OntologyConcept that is provided by an OntologySource. Both OntologySource and DataSource are specializations of Source. Source is managed by the WebServicesAgent and has attributes such as provenance, coverage, access protocol and history. DataSource has attributes such as Quality-of-Service Service-Level-Agreements (QoS-SLAS) and Certificate.

A UserQuery consists of several RefinedQuery, each of which is posed to several DataSource. DataSource provides one-or-more DataItem in response to a RefinedQuery as the QueryResult. Based on the returned QueryResult, the User may provide Feedback as to the result relevance and other comments. These may impact the evolution of metadata associated with UserPreference, query formulation, data source usage and result ranking. A KSMM has been specified by a Protégé ontology [24] and a screen shot of the main panel is shown in Figure 4.

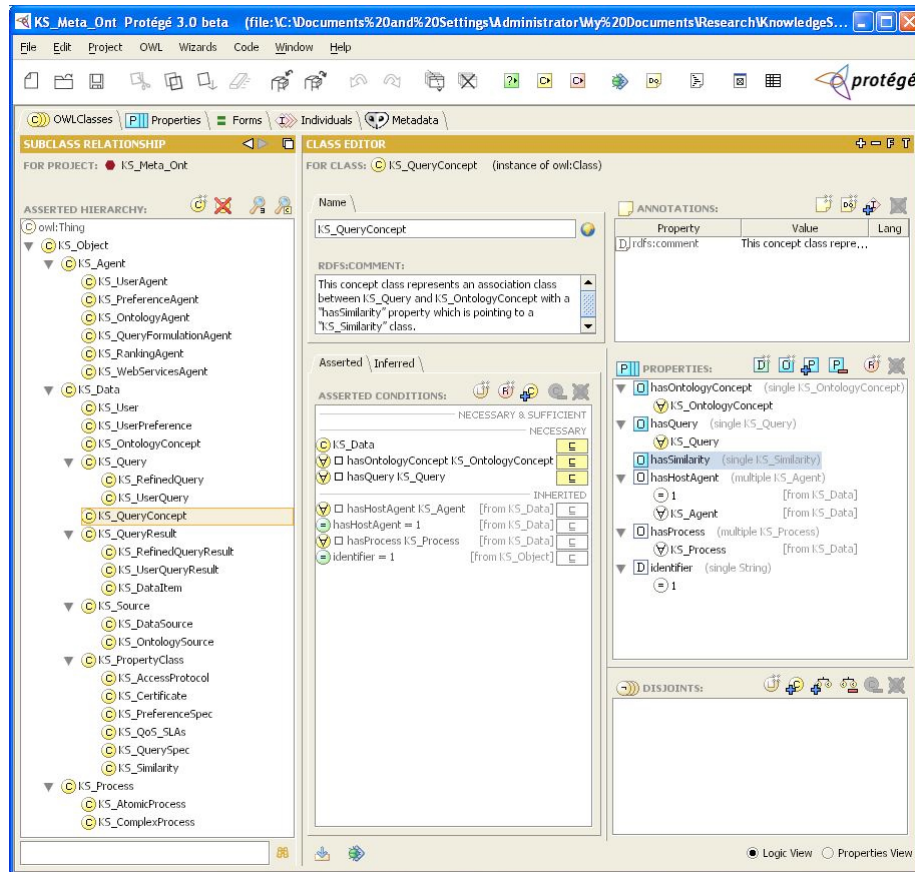


Fig. 4. The Knowledge Sifter Meta-Model in Protégé.

The KS meta-classes correspond to those of the UML diagram in Figure 3. The Protégé KSMM can also be exported to a Web Ontology Language (OWL) specification. This specification can be consulted via a namespace hyper-link, thus

making the agents, which are implemented as Web Services, portable and able to reside on different computers.

3.4 Data/Knowledge Lineage for JIT Customization

The notion of data/knowledge lineage is crucial to the JIT-KM approach. The Knowledge Sifter Meta-Model provides a specification of the object classes, their properties, relationships and constraints. It can also specify workflow processes among the agents that handle user requests and the processing of those requests. This concept can be extended to dynamically configure semantic web services for virtual organizations [8, 9]. In this discussion we focus on how the KSMM can address the just-in-time aspects of Knowledge Management.

By creating the KSMM we can now capture metadata for the overall search process, from the user's initial query specification, to its refinement with semantic ontological concepts, to its processing and ranking. In addition, we can capture agent attributes, measures of agent interaction to determine overall KS performance.

User feedback and KS performance measures and metrics can be used to evolve the system in several ways that affect JIT-KM. For example, User feedback allows the User Preference Agent to adjust the preferences profile to reflect evolving preferences and biases, to adjust the sources that a he prefers and deems to be both of high quality and authoritative. Moreover, as user profiles and preferences are aggregated, we can use data mining and collaborative filtering techniques to discover patterns among groups of users. These learning approaches can be used to make Knowledge Sifter more active in providing JIT-services.

Each Knowledge Sifter Agent can adapt to changing query and web services behavior patterns. For example the User Agent can inform the Web Services and Ranking Agents that a particular user's search and ranking preferences have changed, and that sites such as Google and Yahoo! have emerged as favorites and that their results should receive extra support in the rankings. In addition, the Web Services Agent can monitor network traffic and the response times of data sources to determine whether certain site will not be able to deliver their results in a timely fashion, in which case partial results would be provided in JIT-fashion to the user, until the full results can be assembled.

4 Service-Oriented JIT-Knowledge Management Architecture

This section presents a service-oriented approach to JITKM in which search services such as Knowledge Sifter play a key role. The Knowledge Sifter Meta-Model is used to organize knowledge repository. The overall vision for the JITKM architecture, shown in Figure 5, has three layers: Knowledge Presentation and Creation, Knowledge Management, and Data Sources. At the top layer, knowledge workers may perform searches, communicate, collaborate as well as create and share knowledge. They are provided information by means of the Knowledge Portal, which can be tailored to the profile of each knowledge worker.

The Knowledge Management Layer depicts the Knowledge Repository and the services that are used to acquire, refine, store, retrieve, distribute and present knowledge. These processes are used to create knowledge for the repository. In this paper we are focusing on search services and the just-in-time aspects of KM. One important component of the Knowledge Repository is the Knowledge Sifter Meta-Model, which is instantiated with data from user queries, query processing, the ranked results, and user feedback. In addition, new knowledge can be added to the repository as it becomes available. The services at the Knowledge Management Layer assist in the creation of this knowledge, and its storage in the Knowledge Repository.

The Data Sources Layer consists of the organization's internal data sources including documents, electronic messages, web site repository, media repository of video, audio and imagery, ontology repository and the domain repository, which contains the organization's enterprise model. Also depicted are the heterogeneous external sources of data, including web services that can be used to augment the internal holdings.

Particularly noteworthy are the Ontology Repository and Domain Repository. Organizations may define multiple ontologies used by various divisions in the enterprise, and these organizational ontologies can guide the search process. The Domain Repository contains the Enterprise Model, which is a conceptual model for the organization. It can be used to define the concepts, relationships and constraints that govern the enterprise. For example, in Knowledge Sifter, the ontology sources are WordNet, GNIS, the Imagery Ontology, while the Knowledge Sifter Meta-Model depicts how Knowledge Sifter operates and indicates what data should be captured regarding its processes.

The Knowledge Presentation and Creation Layer

The services provided at this layer enable knowledge workers to obtain personalized information via portals, to perform specialized search for information, to collaborate in the creation of new knowledge, and to transform *tacit knowledge* into *explicit knowledge* [23] via discussion groups. Our work on both WebSifter and Knowledge Sifter has shown that personalized search preferences together with user-specified, ontology-directed search specification and results evaluation can enhance the precision of documents returned by search engines. Thus search services are an important component of knowledge management.

The knowledge creation services, for example, allow knowledge workers to create value-added knowledge by annotating existing knowledge, providing metatags, and aggregating heterogeneous documents into named collections for future use.

Knowledge Management Layer

This layer provides middleware services associated with knowledge indexing and information integration services (IIS). Data warehouse services are listed among the IIS, together with federation, agent, security and mediation services. Services that specifically support JITKM include data mining, metadata tagging, ontology, customization, curation, active updates, QoS-SLAs and workflow services

The next section discusses several services in detail.

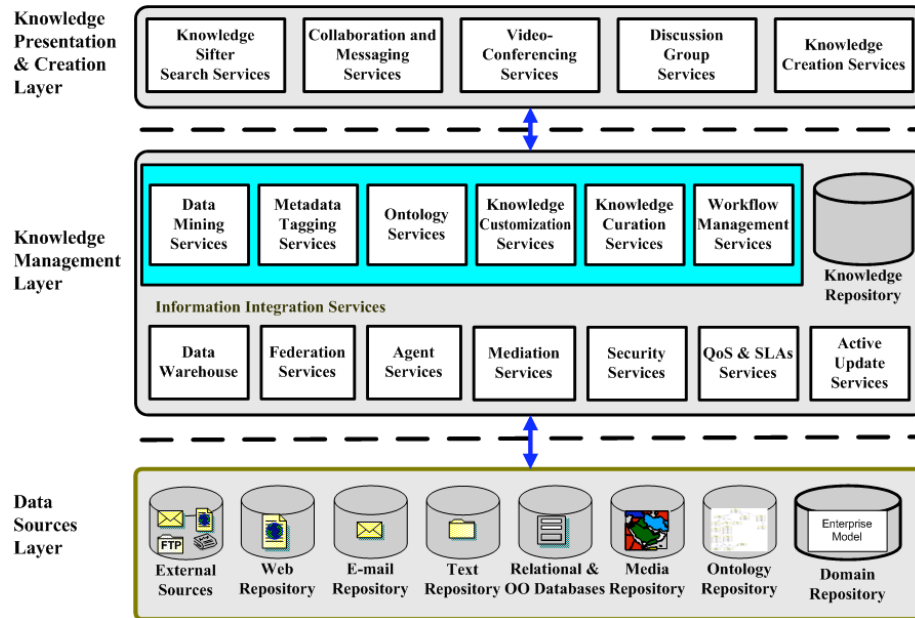


Fig. 5. The Just-in-Time Knowledge Management Architecture

4.1 Services for the JIT Knowledge Repository

Data Mining Services. These services include vendor tools for deducing rules from numeric data, as well as concept mining from text. This knowledge can be used to enhance the Knowledge Repository and to provide refined knowledge to decision-makers. Experience with Knowledge Sifter indicates that understanding the user is crucial to being able to provide relevant search results, based on user intent and context. Mining such information from multiple user queries, and across several users, can improve the user preference model. The research in collaborative filtering can be helpful in suggesting to users other sources of information and related users having similar interests.

Meta-tagging Services. Appropriate indexing of knowledge assets is crucial as collections grow. XML and Resource Description Framework (RDF) are emerging as open standards for tagging and metadata descriptions [19]. The Digital Library community has proposed the Dublin Core Metadata Initiative [3] for tagging books. The tagging of data from heterogeneous sources from the data sources layer is important to allow information to be associated with the concepts defined in the Enterprise Model. Manual tagging is simply too slow for JIT-KM requirements and automated tool and techniques are needed. This is an active research area and a more detailed discussion can be found in [1].

Ontology Services. The construction of domain-specific ontologies is of utmost importance to providing consistent and reliable terminology across the enterprise. Hierarchical taxonomies are an important classification tool, and the trend to use automated tools to specify them [24]. Our research in this area includes the Intelligent Thesaurus [17] and we have used user-specified taxonomies to guide the WebSifter meta-search engine. The intelligent thesaurus is an active data/knowledge dictionary capable of supporting multiple ontologies to allow users to formulate and reformulate requests for information. The intelligent thesaurus is similar to the thesaurus found in a library; it assists analysts in identifying similar, broader or narrower terms related to a particular term, thereby increasing the likelihood of obtaining the desired information from the information sources. In addition, active rules and heuristics may be associated with object types as well as their attributes and functions. This has been used very effectively in our work on WebSifter and Knowledge Sifter. Some of services needed for JIT-KM include ontology mapping, curation, inter-operation, reasoning and merging. Our recent work on the MAKO framework [22] indicates that XML Topic Maps can be used define multiple complementary ontologies that can be merged and used to search a knowledge repository.

Agent Services. The Knowledge Sifter prototype is implemented using agents for each major component. Each agent is implemented as a Web Service and they use standard protocols for communication. Agents exhibit autonomy and are proactive, capabilities that are crucial to JIT-KM. Agents should monitor user behavior to assess not only *intent*, but also the *context* for a user query. Agents can cooperate to solve-problems, to access relevant information from data sources, to package information for decision-makers. Other agents, call staff agents, can monitor the performance of the overall system and provide feedback to line agents to have them adapt to changing behavior and usage patterns.

Mediation Services. Mediation refers to a broad class of services associated with the Intelligent Integration of Information (I*3). Mediation services [28, 31] facilitate the extraction, matching, and integration of data from heterogeneous multi-media data sources such as maps, books, presentations, discussion threads, news reports, e-mail, etc.

Knowledge Customization Services. These JIT services access, customize, package and deliver value-added and focused “knowledge nuggets” to decision-makers. The services consult the User Agent to obtain user preferences, biases and context, including how the user prefers to have information delivered and presented. For example, the curator of the knowledge repository might want to obtain a report regarding the number of users who requested information on ‘Rushmore’ as well as the associated ontological concepts used in query reformulation. This report would be prepared for the curator who might then update the ontology. Another user might want to obtain meta-data associated with the data lineage of his query, including the workflow scheduling, the sources queried, their responses and the results presented.

The user might also wish to peruse those results that ranked lower, but might still be relevant to the task at hand.

QoS & SLAs Services. Quality of Service is an important feature that needs to be considered to retrieve the data from heterogeneous distributed environments. For instance, availability, response time, and throughput of data providers should be measured to optimize resources and provide users data in a timely fashion. These services could monitor the QoS of the data providers and adapt the Web services workflow to compensate for QoS problems. In addition, these services would manage the longer-term relationship using Service Level Agreements (SLAs) [21]. For example, QoS measures could be used by the Knowledge Sifter Web Service Agent to schedule queries, find alternative sources if a data source disconnects from the network, and devise ways to deliver partial information to decision-makers, rather than waiting for all subqueries to finish processing.

Active Update Services. These services ensure that the JIT-KM knowledge repository is up-to-date with respect to JIT requirements. For example, users could create subscriptions to have alerts via e-mail regarding a new paper published by a respected colleague, or the fact that an important event has taken place. User could also pose 'standing queries' that would be run periodically to look for new and interesting results. These queries could be mediated [28] using approximate consistency criteria based on temporal and spatial constraints.

Another agent could examine RSS News Feeds and filter them based on topic and keywords. These could be presented to the user in a newsreader. The agent could also monitor newly tagged messages from the Metadata Tagging Services and present those pertinent to users' context and scenarios. Our research on MAKO [22] indicates how user scenarios, specified as XML Topic Maps, can be shared among users who are collaborating on a problem.

Discovery agents could be tasked to seek out new data sources with information relevant to enterprise needs. They could obtain the Web Service WSDL specification, probe the data source with prototype queries, and assess the overall data quality and the reliability of the source. If accepted, the new source would provide its ontology and the Ontology Services would incorporate the ontology into the Enterprise Model.

5 Conclusions

This paper has presented the concept of Just-in-Time Knowledge Management (JIT-KM) and has discussed a service-oriented Knowledge Management Architecture, together with a collection of services needed to support and realize JIT-KM.

One very important aspect of JIT-KM is to understand the user's information needs, his preferences, biases, and decision-making context. These help to package information into actionable knowledge.

One example of this framework is Knowledge Sifter, the search service of the JIT-KM Architecture. The agents that constitute Knowledge Sifter accept a user's initial query, consult the User Preferences Agent, reformulate the query based on user

preferences and ontological concepts, decompose it into subqueries handled by the Web Services Agent, and rank the query results according to user preferences, biases and context.

The Knowledge Sifter Meta-Model (KSMM) is a specification of the agents, activities, and communications of the system's operation. This has been specified in Protégé, which automatically generates an OWL specification that can be shared with other services via a namespace. The KSMM is one of many holdings of the JIT-KM Knowledge Repository. The KSMM schema can be instantiated with actual user queries, their reformulations, the query processing strategies, the Web Services invocations, the result sets, the rankings, and user feedback regarding the relevance of the results to the task at hand.

In future research we intend to implement the Knowledge Sifter Knowledge Repository based on the KSMM. We hope to use Protégé as a system component to assist the Ontology Agent. Protégé plug-ins such as JessTab and owlTab can be used to provide reasoning and semantic web support, respectively.

Acknowledgements. This work was sponsored by a NURI from the National Geospatial-Intelligence Agency (NGA). The authors wish to acknowledge the work on the Knowledge Sifter prototype by M. Chowdhury, A. Damiano, S. Mitchell, J. Si, and S. Smith.

References

1. Cheng, J., Emami, R., Kerschberg, L., Santos, E., Jr., Zhao, Q., Nguyen, H., Wang, H., Huhns, M., Valtorta, M., Dang, J., Goradia, H., Huang, J. and Xi, S., OmniSeer: A Cognitive Framework for User Modeling, Reuse of Prior and Tacit Knowledge, and Collaborative Knowledge Services. in *Hawaii International Conference on Systems Science (HICSS-38)*, (Island of Hawaii, 2005).
2. Chinnici, R., Gudgin, M., Moreau, J.-J. and Weerawarana, S. Web Services Description Language (WSDL) Version 1.2 (<http://www.w3.org/TR/wsdl12/>), W3C, 2002.
3. DublinCore. Dublin Core Metadata Element Set, Version 1.1: Reference Description, 2003.
4. Fensel, D. Ontology-Based Knowledge Management *IEEE Computer*, 2002, 56-59.
5. Finin, T., Fritzson, R., McKay, D. and McEntire, R., KQML as an Agent Communication Language. in *International Conference on Information and Knowledge Management (CIKM-94)*, (1994), ACM Press.
6. Foster, I., Kesselman, C. and Tuecke, S. The Anatomy of the Grid: Enabling Scalable Virtual Organizations.
7. Hayes, C.L. What Wal-Mart Knows About Customer' Habits, The New York Times, New York, November 14, 2004.
8. Howard, R. and Kerschberg, L. A Framework for Dynamic Semantic Web Services Management. *International Journal of Cooperative Information Systems, Special Issue on Service Oriented Modeling*, 13 (4).
9. Howard, R. and Kerschberg, L., A Knowledge-based Framework for Dynamic Semantic Web Services Brokering and Management. in *International Workshop on Web Semantics - WebS 2004*, (Zaragoza, Spain, 2004).
10. Huhns, M. Agents as Web Services *IEEE Internet Computing*, July/August 2002.
11. Kerschberg, L. (ed.), *Knowledge Management in Heterogeneous Data Warehouse Environments*. Springer, Munich, Germany, 2001.

12. Kerschberg, L. The Role of Intelligent Agents in Advanced Information Systems. in Small, C., Douglas, P., Johnson, R., King, P. and Martin, N. eds. *Advanced in Databases*, Springer-Verlag, London, 1997, 1-22.
13. Kerschberg, L., Chowdhury, M., Damiano, A., Jeong, H., Mitchell, S., Si, J. and Smith, S. Knowledge Sifter: Agent-Based Ontology-Driven Search over Heterogeneous Databases using Semantic Web Services. in Bouzeghoub, M., Goble, C., Kashyap, V. and Spaccapietra, S. eds. *Semantics for a Networked World, Semantics for the Grid Databases, LNCS 3226*, Springer, Paris, France, 2004, 278-295.
14. Kerschberg, L., Chowdhury, M., Damiano, A., Jeong, H., Mitchell, S., Si, J. and Smith, S., Knowledge Sifter: Ontology-Driven Search over Heterogeneous Databases. in *SSDBM 2004, International Conference on Scientific and Statistical Database Management*, (Santorini Island, Greece, 2004), IEEE.
15. Kerschberg, L., Kim, W. and Scime, A., Intelligent Web Search via Personalizable Meta-Search Agents. in *International Conference on Ontologies, Databases and Applications of Semantics (ODBASE 2002)*, (Irvine, CA, 2002).
16. Kerschberg, L., Kim, W. and Scime, A. A Semantic Taxonomy-Based Personalizable Meta-Search Agent. in Truszkowski, W. ed. *Innovative Concepts for Agent-Based Systems*, Springer-Verlag, Heidelberg, 2003, 3-31.
17. Kerschberg, L. and Weishar, D. Conceptual Models and Architectures for Advanced Information Systems. *Applied Intelligence*, 13 (2), 149-164.
18. Kim, W., Kerschberg, L. and Scime, A. Learning for Automatic Personalization in a Semantic Taxonomy-Based Meta-Search Agent. *Electronic Commerce Research and Applications (ECRA)*, 1 (2).
19. Klien, M. XML, RDF, and relatives *IEEE Intelligent Systems*, 2001, 26-28.
20. McIlraith, S.A., Son, T.C. and Zeng, H. Semantic Web Services *IEEE Intelligent Systems*, 2001, 46-53.
21. Menascé, D.A. QoS Issues in Web Services *IEEE Internet Computing*, 2002.
22. Morikawa, R. and Kerschberg, L., MAKO: Multi-Ontology Analytical Knowledge Organization based on Topic Maps. in *Fifth International Workshop on Theory and Applications of Knowledge Management*, (Zaragoza, Spain, 2004).
23. Nonaka, I.O. and Takeuchi, H. *The knowledge-creating company : how Japanese companies create the dynamics of innovation*. Oxford University Press, New York, 1995.
24. Noy, N.F., Sintek, M., Decker, S., Crubezy, M., Ferguson, R.W. and Musen, M.A. Creating Semantic Web contents with Protege-2000 *IEEE Intelligent Systems*, 2001, 60-71.
25. ObjectManagementGroup. Meta Object Facility (MOF) Specification Version 1.3, 2000.
26. Paolucci, M. and Sycara, K. Autonomous Semantic Web Services *IEEE Internet Computing*, Sept - Oct 2003, 34-41.
27. Pouchard, L., Cinquini, L., Drach, B., Middleton, D., Bernholdt, D.E., Chanchio, K., Foster, I.T., Nefedova, V., Brown, D., Fox, P., Garcia, J., Strand, G., Williams, D., Chervanek, A.L., Kesselman, C., Shoshani, A. and Sim, A., An Ontology for Scientific Information in a Grid Environment: the Earth System Grid. in *CCGRID 2003*, (2003), 626-632.
28. Seligman, L. and Kerschberg, L. A Mediator for Approximate Consistency: Supporting 'Good Enough' Materialized Views. *Journal of Intelligent Information Systems*, 8 (3). 203 - 225.
29. USGS. USGS Geographic Names Information System (GNIS), <http://geonames.usgs.gov/>.
30. W3C. OWL Web Ontology Language Overview, <http://www.w3.org/TR/owl-features/>. McGuinness, D.L. and van Harmelen, F. eds., W3C, 2003.
31. Wiederhold, G. Intelligent integration of information. *Journal of Intelligent Information Systems*, 6 (2/3). 203 p.