Rapid Creation and Deployment of Communities of Interest Using the CMap Ontology Editor and the KAoS Policy Services Framework

Andrzej Uszok[1], Jeffrey M. Bradshaw[1], Tom Eskridge[1], and James Hanna[2]

[1] Florida Institute for Human and Machine Cognition (IHMC),
40 S. Alcaniz, Pensacola, FL 32502, USA
{auszok,jbradshaw,teskridge}@ihmc.us
[2] Air Force Research Laboratory, Rome, NY
James.Hanna@rl.af.mil

Abstract. Sharing information across diverse teams is increasingly important in military operations, intelligence analysis, emergency response, and multi-institutional scientific studies. Communities of Interest (COIs) are an approach by which such information sharing can be realized. Widespread adoption of COIs has been hampered by a lack of adequate methodologies and software tools to support the COI lifecycle. After describing this lifecycle and associated dataflows, this article defines requirements for tools to support the COI lifecycle and presents a prototype implementation. An important result of our research was to show how consistent use of ontologies in the COI support tools could add significant flexibility, efficiency, and representational richness to the process. Our COI-Tool prototype supports the major element of the COI lifecycle through graphical capture and sharing COI configurations represented in OWL through the IHMC CMap Ontology Editor (COE), facilitation of the COI implementation through integration with the AFRL Information Management System (IMS) and IHMC's KAoS Policy Services framework, and the reuse of COI models. In order to evaluate our tools and methodology, an example METOC (weather) community of interest was developed using it.

Keywords: Community of Interest, Ontology, COI, OWL, Concept Maps, Policy, KAoS.

1 Introduction

Communities of Interest (COIs) are a realistic approach to interoperability and collaboration on data sharing across organizations that provide an alternative to global data element standardization. They are a means by which the strategy of net-centric information sharing between individuals and organizations can be realized. COIs are defined as "collaborative groups of users who must exchange information in pursuit of their shared goals, interests, missions, or business processes and who therefore must have shared vocabulary for the information they exchange"[3]. Primarily, COIs

F. Zavoral et al. (Eds.): NDT 2010, Part I, CCIS 87, pp. 451–466, 2010.
© Springer-Verlag Berlin Heidelberg 2010

promote data understandability through shared vocabularies, comprised of semantic artifacts (dictionaries, data models, taxonomies, ontologies, etc.), which help members establish a shared understanding of the data they exchange. But they also share policies controlling access to exchanged information and putting obligations on the community members. In any COI, there are two mains roles of participants: a producer and a consumer of information. In any practical community its members usually share these two roles, producing certain information and consuming others. The life of a COI extends beyond the phase of agreeing on vocabulary and policies when, based on abstract definitions, a concrete community is established. All these processes can be significantly augmented by software tools speeding the process of COI creation. This is essential in military, business and scientific domain.

To better understand the needs and requirements of COIs, we began by surveying and assembling a collection of COI resources, which we catalogued on our COI devoted Web site[1]. From these resources and from discussions with a former Commander at the Naval Oceanographic Office, who has extensive experience in the creation of METOC COIs, we derived the necessary requirements for our prototype tool to support the COI lifecycle.

This article presents a description of the Community of Interest lifecycle and dataflow, defines requirements for a tool supporting the COI lifecycle and presents its prototype implementation. Specific objectives for the COI-Tool prototype include the following:

- Ability to model specific roles and physical resources for COI data producers and consumers so that each participant knows what is expected from them;
- Ability to define the semantics and structure of the COI information in a way that allows automatic encoding in a formal, computer processable notation;
- Minimization of training and support requirements for COI managers through simple user interfaces and automation of tedious aspects of lifecycle support;
- Mapping and translation of simple incompatibilities between semantic information from different sources.

In order to address the semantic aspects of COI modeling, we based our approach on the most widely-adopted standard today for semantically-rich model representation - the W3C's Ontology Web Language OWL [11]. We also for the first time integrated two existing IHMC tools CMap Ontology Editor (COE) and KAoS Policy Service with the AFRL Information Management Systems in order to create a comprehensive environment supporting the COI lifecycle. Numerous new mechanisms and capabilities were added to these tools to make this possible.

This paper is organized as follows. First we summarize the key requirements of any tool to support creation and maintenance of COIs (Section 2). We then present the systems we used to create our prototype of the COI-Tool (Section 3). This is followed by the description of the COI-Tool that we have created (Section 4). The next section 5 presents the example weather-related COI which we created by using our new tool. Finally, we conclude the paper with a discussion and a summary.

[1] `http://ontology.ihmc.us/COI`

2 Requirement Analysis of Communities of Interest Lifecycle

The results of our analysis of information flow in the process of creating, implementing and operating the COIs are presented on Fig. 1. First, the COI Manager may use different sources of information to bootstrap the process of defining the initial model. Combining specific data schemas and vocabulary pulled from the existing metadata or ontology repositories (like for instance from the DoD Metadata Registry[2]) as well as saved data from previous COIs and results from the Web and WordNet[3] searches, the initial community model can be rapidly assembled from reusable components and extended to meet new requirements. After this initial step, all the community members have to be involved in the collaborative shaping of the community vocabulary. Still the model at this stage does not usually represent any concrete COI but rather a *reusable template of the given type of community*. For instance, there are many METOC (meteorological) communities that share the same or similar roles and weather data products. Defining a generic template for the METOC community and only refining it for specific case is a time saver.

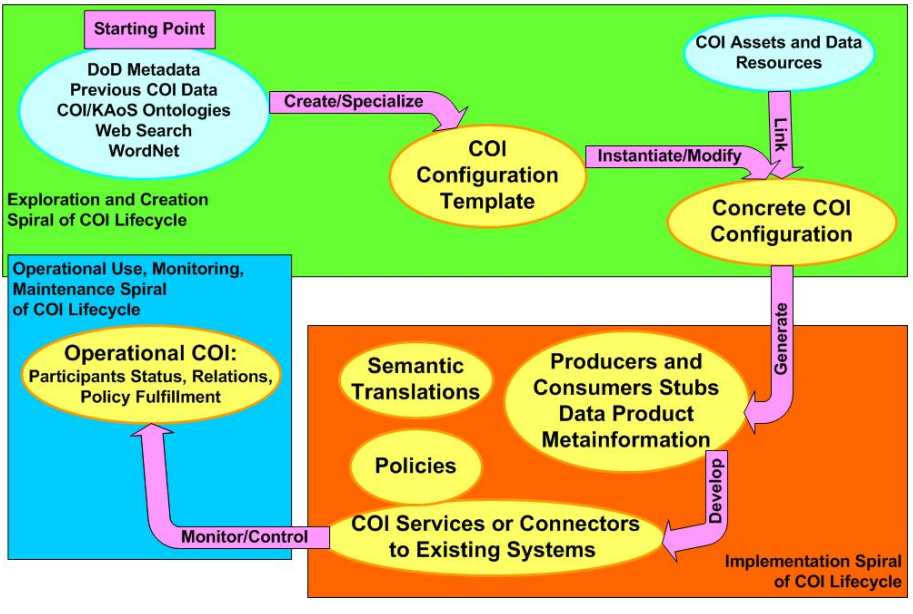

Fig. 1. Data products and dataflow of COI lifecycle

The resulting COI Configuration Template contains the following information:

- Links to ontologies that specify vocabulary for definition of roles and data product structures,
- Specific producers and consumer types,

[2] https://metadata.dod.mil
[3] http://wordnet.princeton.edu/

- Detailed structure of data products,
- Policies defining authorization and obligation,
- Additional specific resources such as maps, images, documents, etc.

From the COI Configuration Template, a concrete COI Configuration can be created by mapping producer and consumer roles from the given COI template to concrete resources such as organizational units, databases, devices, and so forth.

Once the available resources have been mapped to assigned roles, the Implantation Phase of the COI can be started. Software stubs for producers and consumers can be generated based on the defined products data structures. The developer responsible for a particular consumer or producer will use stubs generated by a COI Manager either to implement new services or connectors to existing ones. In this phase, additional details must be decided and implemented. For instance, a COI Manager can refine (add/change/remove) generic policies originally defined as part of the generic COI Configuration Template model. If required, simple semantic translations can be also defined in order to harmonize data from incompatible participants.

Finally in the next phase, the members can be activated inside the community infospace. A COI Manager should be able to monitor the community and obtain information about participant status, relations between them and compliance with the policies.

In the next three subsections we present the detail requirements, which are the results of our examination, for any practical tool supporting the COI lifecycle.

2.1 Exploration and Creation Phase Requirements for the COI-Tool

During the exploration and creation phase, the COI-Tool has to support the COI Manager and the participants of the community in the definition of a model for the COI. The important part of this process is an agreement on the terms and vocabularies to be used to exchange information products. In order to support this in an effective way, the following three main requirements have to be fulfilled:

- *Easy-to-understand formal models of COI information representation* - COI-Tool must capture the COI requirements and structure in an easy-to-understand format that can be automatically transformed into a formal model of COI information requirements.
- *Support for collaborative COI development* - to facilitate the process of achieving consensus on the COI vocabulary, roles and information to be exchanged, the COI-Tool should provide support for both synchronous and asynchronous distributed collaboration on vocabulary creation for the COI participants.
- *Ease of reuse* – COI-Tool should possess graphical templates for individual model elements or larger model structures so they can be easily extended and reused, thus saving development time in similar COI applications that may later arise.

2.2 Implementation Phase Requirements for the COI-Tool

This phase of the COI lifecycle involves physical implementation of a concrete community. Support for this phase must allow mapping abstract elements of a COI model to physical assets (e.g., organizations, individuals) and data resources. Software components must be implemented or adapted to connect the required resources of producers and consumers to the community infospace. Additionally, meta-information for community data products has to be easily generated from the model. In particular the following requirements have to be fulfilled by the tool:

- *Link abstract COI model to implementation model* - COI-Tool should provide a graphical interface to support mapping. Once the mapping is complete the tool should automatically generate the configuration files and software stubs that will be used to implement producers and consumers.
- *Data product policies.* COIs need an expressive and flexible way to define data product access and filtering policies, as well obligation policies for members.
- *Harmonization of vocabulary.* Finally, any realistic COI will need a way to accommodate at least simple differences in vocabulary among partners within the COI model.

2.3 Operation, Monitoring and Maintenance Phase Requirements for the COI-Tool

During the Operational phase the tool should provide the following capability to a COI Manager:

- *Monitoring configuration, activity state, and policy compliance* - authorized COI participants must be able to monitor activities inside the community, to discover discrepancies between the model and reality, and to be notified if community policies are violated.
- *Monitoring producer/consumer/information object relationships* - allowing to asses the state of the community and compliance with policy.
- *Collecting overall history and statistics* – essential for any forensic investigation.

3 COI-Tools Prototype as an Integration of Existing Systems

During the development phase, our objective was to build and test a prototype implementation of the requirements enumerated during the analysis phase of the project. We found that existing IHMC and AFRL systems, when combined and enhanced in specific areas, would fulfill the requirements for COI-Tools.

Fig. 2 maps the requirements for the COI Tool described and justified in the previous section to existing or new features of the COE, KAoS and IMS components. Below we briefly describe these systems.

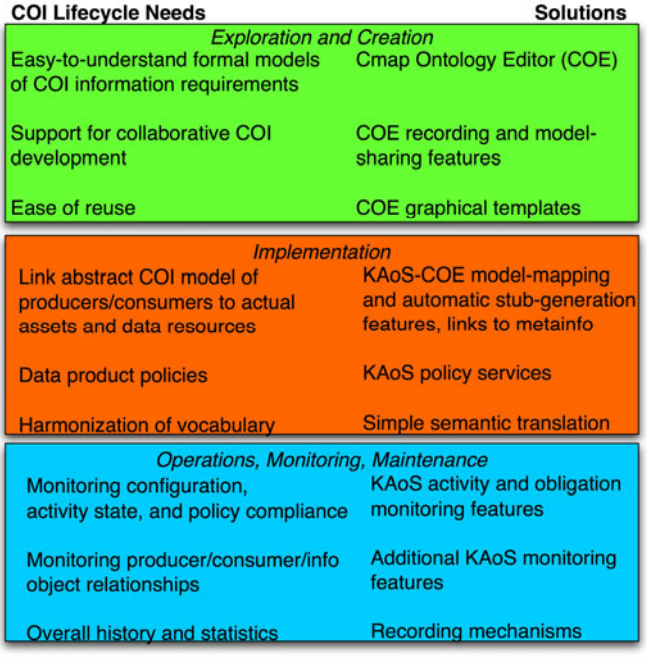

Fig. 2. Summary of COI Lifecycle Needs and Solutions

3.1 COE - CMap Ontology Editor

COE[4] [4,8] is an extension of the IHMC CMapTools[5] concept mapping program [2]. COE allows OWL ontologies to be conceptualized, developed, and managed though a powerful graphical interface based on concept maps. Unlike other tools [5,9], COE was specifically developed in order to exploit patterns of OWL structure to make it easier for both experts and non-specialists to use (e.g., hiding irrelevant information, use of templates for frequently-used patterns). To bridge the gap between the informal nature of concept maps and the formal, machine-readable Web ontology languages, COE uses a set of conventions and guidelines that enables users to construct syntactically valid Web ontologies using the concept-mapping interface. These conventions retain as far as possible an intuitive reading of the concept map while faithfully capturing the precision of the OWL syntax, and are based on a few basic ideas (which make them easy to learn). English words and phrases are used as far as possible, and we avoid the 'mathematical logic' terminology that pervades the OWL documentation.

Figures 7 and 8 shows examples of ontologies depicted using COE, where the blue lines denote class inclusion, and the dotted lines denote property definitions. The text boxes containing unprefixed labels are defined as part of the ontology under development (e.g., "ForecastElement"), while the prefixed labels are concepts that are linked in from other existing ontologies (e.g., "time:ProperIntervalThing"). Linking

[4] `http://cmap.ihmc.us/coe`
[5] `http://www.cmappers.net/`

our specific ontology elements to existing ontologies is a central aspect of our approach, whereby we reuse existing concepts from other ontologies and thereby achieve interoperability simply by being written in the same language. The side panel shown in Fig. 7 shows the list of ontologies that are being imported, and provides several different ways to browse and connect their information.

3.2 KAoS Policy Services Framework

IHMC's KAoS[6] [10] is a mature services framework that relies on ontology in the specification, analysis, and enforcement of policy constraints across a wide variety of distributed computing platforms. KAoS Policy Services allow for the specification, management, conflict resolution, and enforcement of policies. KAoS policies distinguish between authorizations (i.e., constraints that permit or forbid some action by an actor or group of actors in some context) and obligations (i.e., constraints that require some action to be performed when a state- or event-based trigger occurs, or else serve to waive such a requirement).

The use of ontology to represent policies enables reasoning about the controlled environment, about policy relations and disclosure, policy conflict resolution, as well as about domain structure and concepts. KAoS reasoning methods exploit description-logic-based subsumption and instance classification algorithms and, if necessary, controlled extensions to description logic (e.g., role-value maps).

Two important requirements for the KAoS architecture have been modularity and extensibility. These requirements are supported through a framework with well-defined interfaces that can be extended, if necessary, with the components required to support application-specific policies. The basic elements of the KAoS architecture are shown in Fig 3; its three layers of functionality correspond to three different policy representations:

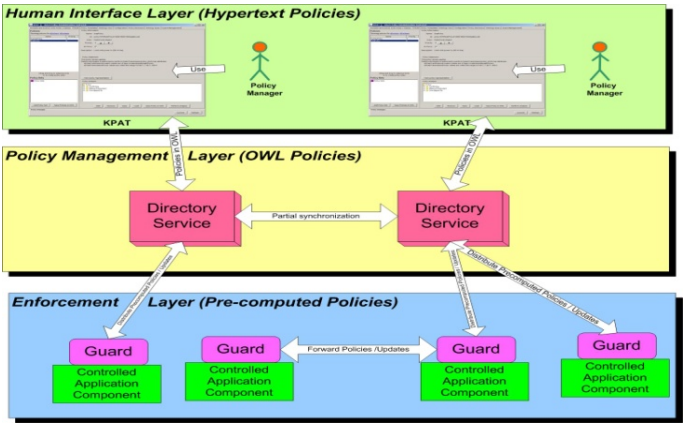

Fig. 3. KAoS Policy Services Conceptual Architecture

[6] http://ontology.ihmc.us/kaos

Human interface layer: This layer uses a hypertext-like graphical interface (KAoS Policy Administration tool, or KPAT[7]) for policy specification in the form of natural English statements. The vocabulary for policies is automatically provided from the relevant ontologies. Management of KAoS components and of the ontologies can also be performed by authorized users through KPAT. In addition to the standard KPAT interface, customizable policy templates and wizards make policy creation easy even for non-specialists.

Policy Management layer: Within this layer, OWL is used to encode and manage policy-related information. The KAoS Distributed Directory Service (DDS) residing in this layer encapsulates a set of ontology reasoning mechanisms.

Policy Monitoring and Enforcement layer: KAoS automatically "compiles" OWL policies to an efficient format that can be used for monitoring and enforcement, and handles selective distribution of policies and ontological concepts to the KAoS (software) Guards that need them. This compact policy representation provides the grounding for abstract ontology terms, connecting them to the instances in the runtime environment and to other policy-related information. KAoS Guards interface with the applications being governed by policy. The guards provide a well-defined API to support policy decisions and information queries. The "pre-computed" nature of the guards' policy representation and the simplified reasoning engine means that policy decisions can be rendered locally with "table-look-up" efficiency. The network connection to the DDS is needed only if and when runtime policy updates are made.

Although the description-logic reasoning integrated with KAoS handles most requirements, specialized reasoners (e.g., spatial and temporal reasoning, probabilistic reasoning) can be accommodated as "plug-ins" to KAoS to meet application-specific needs.

3.3 AFRL Information Management System

The Information Management System developed at AFRL [6] utilizes the concept of an infosphere that includes many diverse highly distribute applications operating on data (called clients) and a set of core services that enable the dissemination, persistence and control of information being shared among these applications. Key services provided to disseminate information are

- publish and subscribe – allowing a consumer to register a subscribe to a specify type of information and obtain these type of information from any publisher in the system.
- query - allow a consumer to query for specific type of date fulfilling specific predicate the archive of previously published information.

The quantum of managed information is called a managed information object (MIO). A MIO comprises a payload and metadata that characterizes the object (e.g., topic, time, and location, in general XML Schema). It is desirable that all of the information needed for making information management decisions (such as content-based brokering and dissemination) be present in the metadata in a form that permits efficient

[7] Pronounced "KAY-pat."

processing. An important message characterization element is the concept of "type" (e.g., "satellite imagery"). Type is used to determine what policies describe its appropriate use. The system's design as a set of independent services [7] allows for integration of a very diverse producers and consumers of information.

4 COI-Tool Architecture and Functionality

The tool's functionality (Fig. 5) uses ontologies as a central component. It was assumed that any model of a community of interest would start from the generic COI ontology developed during the project, which is itself based on the KAoS ontology[8]. Through this extension, any community model can be a source for the vocabulary for the KAoS-defined policies of the community. Our generic COI ontology captures the basic elements of any COI. Using it as a starting point for domain specific COI ontologies establishes common ground in terms of vocabulary and operations that will help standardize practice and enable collaboration across COIs. The tool is further grounded to the specific infosphere technology by using the IMS mapping between the COI ontology and IMS ontology developed in the J-DASP project[9]. The COI-Tool produces further layers of specialization that build on this grounded framework in the form of concrete COI ontologies developed for specific communities (Fig. 4).

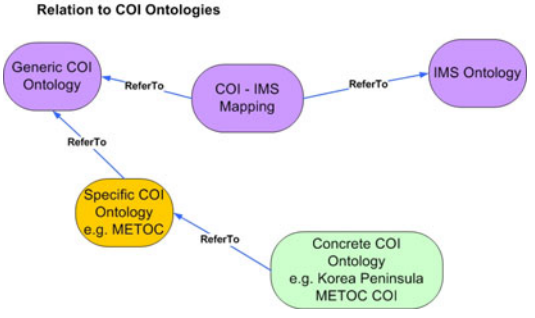

Fig. 4. Developed COI ontologies and their relations

COE is used for almost all the graphical interaction with the user with exception of policy definition, which is done by KAoS KPAT interface. The original CMap Tool on which COE is based allows for easy plug-in extensions with new graphical menus activating specific new actions on the concept maps. We developed new specific extension in order to provide specific COI related menus of actions to COE.

KAoS provides policy, domain and matching services as well as serving as an integration framework for the COI-Tool, conveying messages between COE and IMS clients fulfilling roles in a concrete operating COI. The IMS provides an examples infosphere implementation which can support community communications. However it is possible to replace it with other infosphere technology.

[8] http://ontology.ihmc.us/ontology.html
[9] http://jbi.isx.com/jdasp/

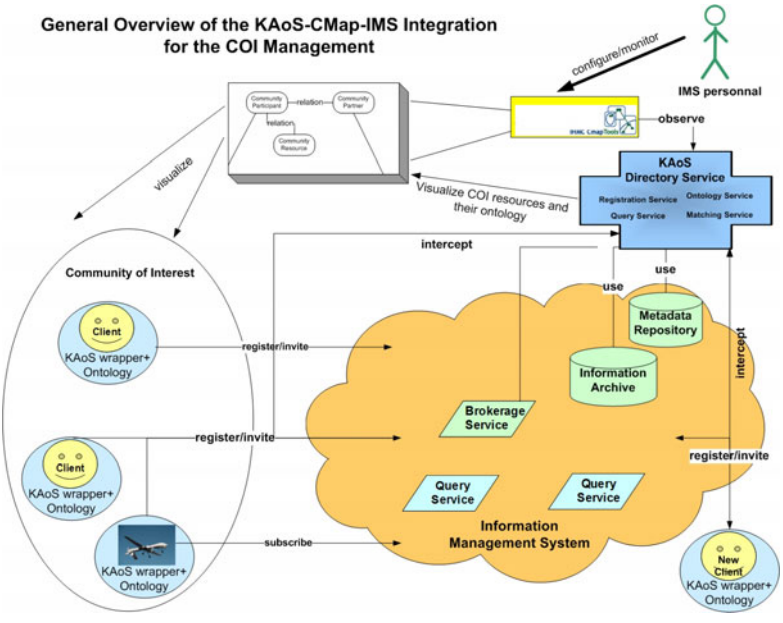

Fig. 5. Overview of the COE-KAoS-IMS integration for COI-Tool

4.1 COI-Tool Functionality for the COI Exploration and Creation Phase

We aimed at providing a rich graphical environment to model the COI Configuration Template and its mapping to a specialized, concrete configuration. Considerable effort was spent on the enhancement of the COE Ontology Editor functionality. We added the ability for COE to edit, browse, and manipulate multiple ontologies, which is an essential requirement for real applications. We also implemented a mechanism to automatically generate OWL encodings of a map.

COI managers are provided with the following functionality when they want to create a new COI, modify an existing one, or collaborate with other community members:

Creation of a new COI
This new COE menus option allows bootstrapping the COI creation process by the generation of four COI configuration concept maps for definitions of community partners (roles/actors), data products, *classes of actions,* and COI properties such as information about the manager's identity, applications used, and so forth. Fig. 6 shows the new menu in COE and the resulted generated maps (on the right in the list of maps).

These new placeholders are empty concept maps dedicated to particular part of the community model. However, they are already linked to the appropriate parts of the COI generic ontology, so the menu of concepts available in the particular maps is focused according to its context. The new COI template can be started from the generic ontology but also from the existing COI templates.

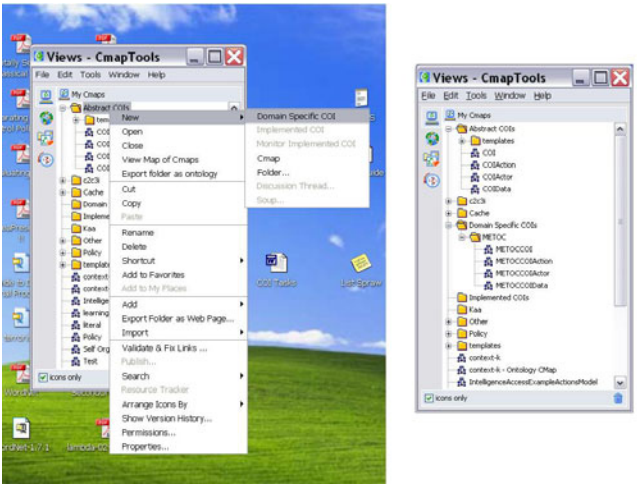

Fig. 6. GUI for creation of new domain specific COI and the result

Definition of COI properties, roles and data product
A COI manager can open any of these maps in the editor and graphically define concepts and their relations. He can use the Semantic Space Panel, developed in the scope of this project, to access concepts defined in other concept maps (ontologies) such as, for instance, the weather ontology when defining weather products (Fig. 7).

The Semantic Space (Namespace) Panel functionality includes the ability to:

- Show a list of concepts, from the selected namespaces, (as a hierarchy with additional information) applicable to a given selected map node,
- Allow the user to drop a selected concept on the map node and automatically create appropriate map semantic constructs,
- Show current semantic information, from the chosen namespaces, about the selected map node,
- Use the ontology reasoner to compute the menu list of concept instances or superordinates based on the selected concept node.

Usage of Web search and WorldNet
The COI-Tool provides access to the existing Web resources in order to obtain concept definitions or related concepts. The found concepts can be added to the map by creation of new map nodes using the button from the COE search window.

Definition of relationships between COI roles and COI products through the use of COE templates
We developed automatic generation of concept map templates to be used in the specialized COI maps. When a given community map is saved, a set of templates is generated from it, making it easy to create subclasses or instances of concepts from this map. So, for instance, it is easy to specify that a given community role is a specialization of some other role because of the existence of the template. Annotation of every map with the parent ontology allows the tool to select which template should be shown for a given map. In addition, there is a set of static templates making creation

of generic ontological relations easy and readily understandable. This set can be easily extended using the COE template editor. The COI-Tool has a special panel on the right dedicated to template management.

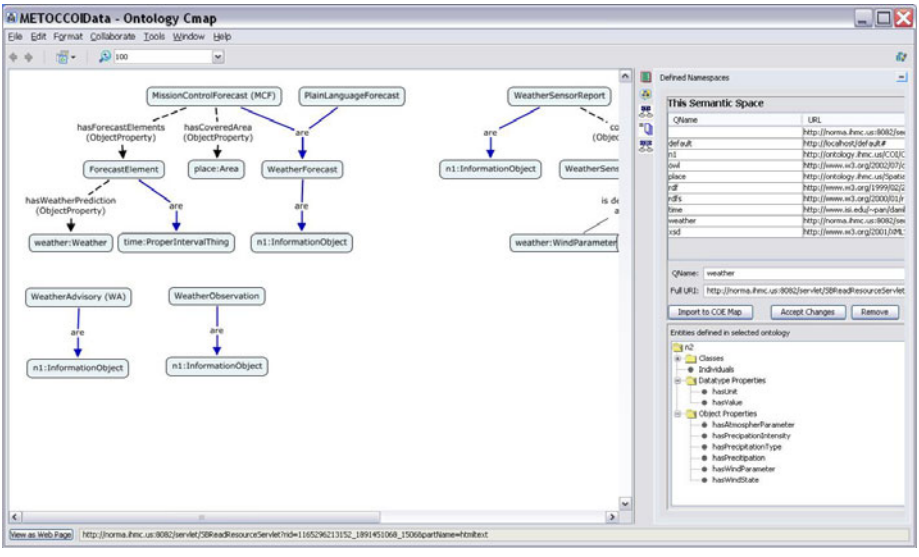

Fig. 7. GUI enabling the modeling of COI roles and products

Usage of collaboration mechanisms
The COI Manager and his partners are able to simultaneously edit concept maps with community vocabulary and models using the COI-Tool. If only their tools are connected to the same CMap server they can invite each other to edit specific maps. If only they accept the invitation, they will see, in their COE window, the edits and annotations performed by others, will be able to edit maps themselves, what will be immediately visible to other members, and will be able to send chat messages to others. This mechanism allows them to archive a consensus on concepts definitions, data products definitions, and so forth.

Addition of links and multimedia resources
A COI manager can in the COI-Tool embellish a COI concept by adding URLs to web resources, links to documents and maps. This additional information provides context and related information important in facilitating the shared human understanding of the created community model.

Access to historical versions of concept maps
COE has been integrated with Subversion[10] to make it possible to store and then subsequently access previous versions of the developed maps from the integrated version repository. The COI-Tool provides a browser for viewing the different revisions of the community concept maps and a compare tool for identifying their differences.

[10] `http://subversion.tigris.org/`

Creation of a new implemented COI configuration
A COI manager can create the implementation concept maps of a given COI Configuration Template using the COI-Tool. The dedicated map for the mapping between resources and community roles has specialized grahical templates making this process easier.

4.2 COI-Tool Functionality for the Implementation Phase

The COI-Tool mechanisms described here are targeted for deployment of a given COI as a set of clients in an infosphere based on the ARFL Information Management System. A similar support can be added to the COI-Tool for other information systems realizing COI.

Generation of bootstrap files and code skeletons
The COI manager can generate a set of configuration and implementation files specifically for each community participant, and delivers them to the developers responsible for that participant. This file set contains only the relevant information for the given participant. Based on this information, the developer can implement a client for the Information Management System infosphere producing and consuming products with the appropriate for its roles types and structure.

Definition of COI policies
The OWL representation of the given COI can be directly used as an ontology vocabulary to define the COI policies in KAoS. Using KAoS is it not only possible to define authorization policies controlling distribution of information, but also obligation policies obliging, for instance requiring the timely manner of issuing periodic updates.

Definition of semantic translation
In reality when implementing COI, some partners will use incompatible representation of data, for instance when coming from some other community of interest. The COI-Tool has been equipped with a prototype graphical interface allowing the mapping of structures between two ontological classes defining schemas for data products. The interface generates translation code in SPARQL[11], which then can be used to translate concrete data during communication.

4.3 COI-Tool Functionality for the Operation, Monitoring and Maintenance Phase

In order to provide monitored and other functionality for the community of interest we have developed and implemented in the special interceptor for the IMS clients. This interceptor modifies the standard IMS client library linked with every client and transparently analysis client communication. The new module provides the following functionality:

- Monitoring of the members status and relation; reports when a given member is activated and what data its produces and from whom it received data,

[11] `http://www.w3.org/TR/rdf-sparql-query/`

- Collects statistical information about the relations; first/last time of data consumption, frequency and average time between consumptions,
- Checking authorization policies compliance,
- Monitoring of obligation policy fulfillment,
- Semantic translation.

Monitoring of partners activation and relations
The information intercepted in the IMS clients is forwarded through KAoS to COE if the map with the community model is opened by its manager. The map will show which participants are active and if any unanticipated clients are present. It will also show by way of linking lines any producer–consumer relation. This links can be clicked to show statistical information.

Monitoring of obligation policies
In order to monitor the fulfillment of the obligations of an information producer, we developed the new KAoS mechanism – the policy monitor, which allows determining compliance with the obligation policies. If any obligation is violated, a feedback call-back to the COI Monitor Mode is activated; a notification is issued and showed up on the appropriate community map.

Recording of monitoring session and access to recorded session
The history of a community session can be stored in a special folder associated with the COI configuration. Any community activity session can be also replayed later.

5 Proof of Concept – METOC Community of Interest

In order to test the usability of the presented methodology and the implemented prototype tool, an example METOC community of interest was developed. Background research for this community and a collected set of references is available on our web site dedicated to METOC (*http://ontology.ihmc.us/coi/metoc.htm*).

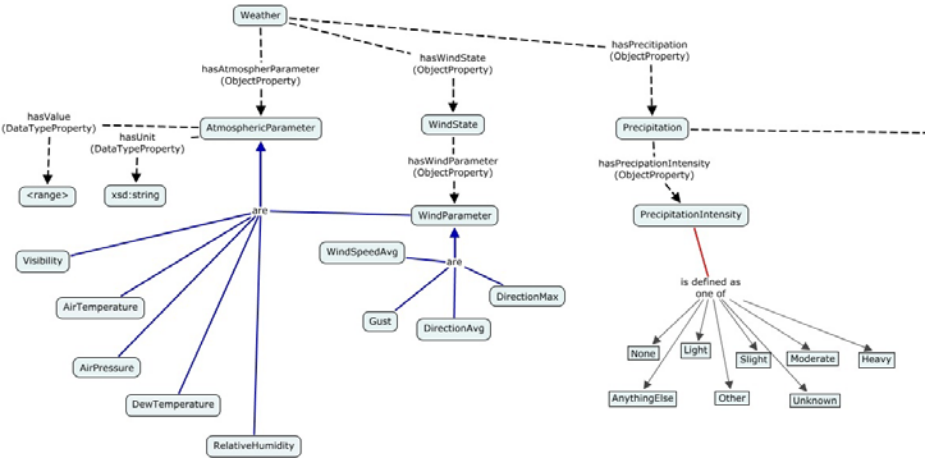

Fig. 8. Fragment of the concept map with vocabulary for weather related concepts

Using the COI-Tool, we developed weather ontology (Fig. 8) and then used it to define different weather data products as we found in the collected resources.

Based on the concrete documents for the Korea METOC Operation Plan, as well as air and space weather operations, we defined roles in this community and then mapped them to the actual units in the Korea military area (Fig 9). The model was then used to generate bootstrap files and stubs for example clients, which were developed in AFRL Information Management System. The clients simulated collection of weather sensor data and their distribution to the weather forecast centers. The centers were obliged by policy to periodically published weather forecast and, if necessary, weather warnings. We defined policies controlling access to the weather sensor data and obligations for providers of data and issuer of forecast and weather warnings.

Fig. 9. Monitoring the Korea METOC community

This use case established the feasibility of COI managers relying on the tool to develop realistic communities of interest and to support all aspects of its lifecycle.

6 Conclusions

The key results of our research include a deepened understanding of the community of interest lifecycle, definition of requirements for tools supporting COI lifecycle and description of the COI lifecycle dataflow. Additionally, we demonstrated how the consistent use of ontologies in the COI supporting tools can add significant flexibility and representational richness to the process.

Our current work with AFRL concentrates on federation mechanism, controlled by policy, for new tactical service-oriented versions of the IMS and it is aimed at supporting even more diverse and dynamically created communities (such as coalitions).

Even though the system was evaluated in a military application, the lessons learned and the tools developed can be used in business, academic, or research domains in order to create common vocabulary/ontology allowing for integration or creation of alliances between people and enterprises. We believe that the system can be useful for the sharing information in the scientific observation and experiments.

Acknowledgements

We are grateful for the contributions and help from of Robert Hillman, Asher Sinclair, Niranjan Suri, James Lott and Maggie Breedy. This research has been funded by the Air Force Research projects (reference FA8750-06-2-0065 and FA8750-07-2-0174).

References

[1] Baader, F., et al. (eds.): The Description Logic Handbook. Cambridge University Press, Cambridge (2003)
[2] Cañas, A.J., Hill, G., Carff, R., Suri, N., Lott, J., Eskridge, T., et al.: CmapTools: A Knowledge Modeling and Sharing Environment. In: Cañas, A.J., Novak, J.D., González, F.M. (eds.) Concept Maps: Theory, Methodology, Technology. Proceedings of the First International Conference on Concept Mapping, vol. I, pp. 125–133. Universidad Pública de Navarra, Pamplona (2004)
[3] DoD Chief Information Officer, DoD Net-Centric Data Strategy (March 2003), http://www.defenselink.mil/nii/org/cio/doc/Net-Centric-Data-Strategy-2003-05-092.pdf
[4] Eskridge, T., Hayes, P., et al.: Formalizing the Informal: A confluence of concept mapping and the semantic web. In: Cañas, A.J., Novak, J.D. (eds.) Concept Maps: Theory, Methodology, Technology. Proceedings of the Second International Conference on Concept Mapping, vol. 1. Universidad de Costa Rica, San Jose (2006)
[5] Fortuna, B., Grobelnik, M., Mladenic, D.: OntoGen: Semi-automatic Ontology Editor. In: Smith, M.J., Salvendy, G. (eds.) HCII 2007. LNCS, vol. 4558, pp. 309–318. Springer, Heidelberg (2007)
[6] Infospherics Web Site, http://www.infospherics.org
[7] Grant, R., Combs, C., Hanna, J., Lipa, B., Reilly, J.: Phoenix: SOA based information management services. In: Proceedings of the 2009 SPIE Defense Transformation and Net-Centric Systems Conference, Orlando, Fl (2009)
[8] Hayes, P., Eskridge, T., et al.: Collaborative knowledge capture in ontologies. In: Proceedings of the 3rd International Conference on Knowledge Capture, Banff, Alberta, Canada. ACM Press, New York (2005)
[9] Sarker, B., Wallace, P., Gill, W.: Some Observations on Mind Map and Ontology Building Tools for Knowledge Management, Ubiquity, vol. 9(9). ACM Press, New York (2008), http://www.acm.org/ubiquity/volume_9/pf/v9i9_sarker.pdf
[10] Uszok, A., Bradshaw, J., Lott, J., Breedy, M., Bunch, L., Feltovich, P., Johnson, M., Jung, H.: New Developments in Ontology-Based Policy Management: Increasing the Practicality and Comprehensiveness of KAoS. In: Proceedings of the IEEE Workshop on Policy 2008. IEEE Press, Los Alamitos (2008)
[11] Web Ontology Language, http://www.w3.org/TR/owl-features