ICOM: A Framework for Integrated Collaborative Work Environments

Kenneth Baclawski[1], Eric Chan[2] Laura Drăgan[3], Patrick Durusau[4], Deirdre Lee[3], Peter Yim[5], and Yuwang Yin[1]

[1] Northeastern University, Boston, MA 02115, USA,
kenb@ccs.neu.edu, ywang.yin@gmail.com,
http://www.ccs.neu.edu/home/kenb
[2] Oracle Corporation, Redwood Shores, CA 94065, USA,
eric.s.chan@oracle.com
[3] Digital Enterprise Research Institute, Galway, Ireland,
firstname.lastname@deri.org
[4] Consultant,
patrick@durusau.net,
http://tm.durusau.net
[5] CIM Engineering, Inc. ("CIM3"), San Mateo, CA 94402, USA,
peter.yim@cim3.com,
http://ontolog.cim3.net/cgi-bin/wiki.pl?PeterYim

Abstract. Collaboration is an important activity that is increasingly using technology to improve the productivity of the participants. The Integrated Collaboration Object Model (ICOM) is a proposed OASIS standard for interoperation among collaboration services. ICOM is intended to be a framework for integrating a broad range of domain models for collaboration environments. The intention is to encourage independent software vendors and open source communities to create common collaboration clients that interoperate with integrated collaboration platforms and standalone collaboration services across enterprise boundaries. This paper provides an overview of ICOM that covers the high-level concepts, directory, space, access control, metadata, content management, and unified message models. ICOM is represented in several formats: the Java Persistence API, XML Schema, RDF and OWL. We also describe an example application based on ICOM.

Keywords: JPA, collaborative work environments, Semantic Web

1 Introduction

The Integrated Collaboration Object Model (ICOM) for Interoperable Collaboration Services defines a framework for integrating a broad range of domain models for collaboration activities in an integrated and interoperable collaborative work environment. The framework is not intended to prescribe how applications or services conforming to its model implement, store, or transport the data for objects. It is intended as a basis for integrating a broad range of collaboration objects to enable seamless transitions across collaboration activities. This enables applications to maintain a complete thread of conversations across multiple collaboration activities. The model integrates a broad range of collaboration activities, by encompassing and improving on a range of models which are part of existing standards and technologies. The model is modular to allow extensibility. The fundamental concepts, metadata concepts, and their relations are included in the core part of the model, while the specific concepts and relations for each area of collaboration activities are defined in separate extension modules.

In the next section, we define the ICOM Core model. It defines the classes that bring together the model of directory management, identity management, and content management in a framework with a common access control model and metadata model. We continue with the description of the extension modules, which extend the core to define the specialized models for different collaboration activities. The range of collaboration models include content sharing and collaboration, asynchronous communication, instant communication, presence awareness, moderated group discussion, time management, coordination, etc. The core defines three distinctive branches of concepts, which complement each other: Artifact, Subject, and Scope.

The Subject and Artifact branches support separation of concerns for user administration and content management. The Subject branch includes the model of actors, groups, and roles. These concepts typically appear as the subject in the (subject, privilege, object) triples of an access control model. The Artifact branch includes the model of content and metadata produced by actors.

The Scope branch defines the model of communities and spaces that contain subjects and artifacts. Communities and spaces join the subjects and artifacts in a role-based access control model where a role is assigned to an actor in a specific scope. Thus Scope, Subject, and Artifact form a framework for applications to integrate and interoperate with directory, identity management, content management, and collaboration services.

The model specified in ICOM is based on existing standards and technologies. ICOM core model encompasses LDAP Directory Information Models [1]. The extension modules integrate models from Content Management Interoperability Services [2], Java Content Repository API [3], Web Distributed Authoring and Versioning (WebDAV) [4], Internet Message Access Protocol (IMAP) [5], Simple Mail Transfer Protocol (SMTP) [6], Extensible Messaging and Presence Protocol (XMPP) [7], XMPP Instant Messaging and Presence [8], vCard MIME Directory Profile [9], Internet Calendaring and Scheduling Core Object Specification (iCalendar) [10], and Calendaring Extensions to WebDAV (CalDAV) [11]. ICOM is open for extensions with additional domain models to enable seamless integration with business processes and social networks: for example in process integration domain which includes Business Process Model and Notation [12], Web Services Business Process Execution Language [13], WS-BPEL Extension for People [14], and Web Services for Human Task [15]; in social networking domain, which includes Friend of a Friend (FOAF) [16], Semantically-Interlinked Online Communities (SIOC) [17], Open Social[6], and Facebook Platform Open Graph[7].

ICOM uses the Content Management Interoperability Services (CMIS) [2] grammar to define classes and properties.

The OASIS ICOM TC Wiki[8] provides supplemental information, including overview, primer, extensions, use cases, and mappings to various standards and proprietary data models. The integrated model can be the foundation for defining the application programming interfaces (API) for integrated collaboration applications to interoperate with collaboration services. A service provider interface (SPI) can be specified to support interchangeable and interoperable services that conform to the ICOM application framework. ICOM does not prescribe how applications or services conforming to its model implement, store, or transport the data for objects.

ICOM is represented in several formats: the Java Persistence API, XML Schema, RDF and OWL. The UML and XML Schema are derived from the Java classes, while the other specifications are directly derived from the authoritative specification.

2 ICOM Modular and Extensible Framework

ICOM specifies a set of concepts in a collaboration environment, in terms of class and property definitions. An ICOM object may be composed of information from multiple repositories or collaboration services. All objects in the ICOM framework must be instances of at least one class. The class and property definitions correspond to the UML meta-model, which is an OMG Meta Object Facility (MOF) M2-model. The UML diagrams for ICOM were generated from Java classes, which were directly translated from the authoritative ICOM specification. Some of these diagrams are shown below. The full set of UML class diagrams is in [18].

2.1 ICOM Core

The ICOM Core model has three branches in its class hierarchy: Scope, Subject, and Artifact. The Scope branch includes the model of communities and spaces which are containers of subjects and artifacts. This branch is concerned with directory management, providing hierarchical classified listings of Role, Group, and Actor for administration, search and indexing, and uniform reference. Community and space list the resources as directory entries [1]. Although spaces contain both subjects and artifacts, the membership of subjects in a space is administered separately from management of artifacts in the space. Discretionary

[6] http://opensocial.org

[7] https://developers.facebook.com/docs/opengraph

[8] https://wiki.oasis-open.org/icom

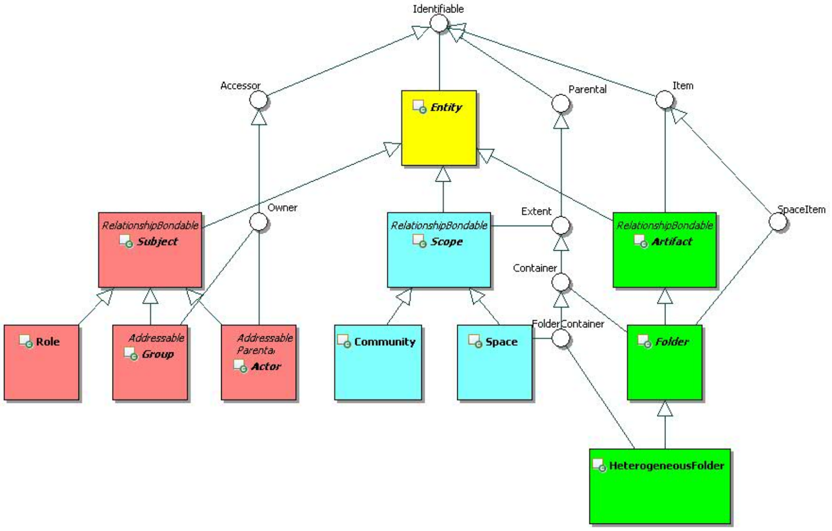


Fig. 1. The Main Branches of the ICOM Core Model

access control (DAC) and role-based access control (RBAC) policies are defined in terms of the subjects from the directory.

The Subject branch includes the model of actors, groups, and roles. This branch is concerned with the identity of actors in collaboration. The Artifact branch includes the model of folder, heterogeneous folder, and content produced by actors.

The Core model also defines the Metadata and Access Control models in separate namespaces. The Metadata is concerned with annotations. The Access Control defines the discretionary access control (DAC) model for entity-level granularity and role-based access control (RBAC) model for scope-level granularity.

The Subject and Artifact branches support the separation of concerns of user administration and content management. Typically subjects and artifacts are joined in the (subject, privilege, artifact) triples of the access control model. Some of the triples are derived from the scopes of the role assignments and the artifacts contained by the scopes.

Figure 1 depicts the top-level abstract classes forming the main branch and the Scope, Subject, and Artifact classes that represent the roots of the three major sub-branches of the ICOM class hierarchy. To deal with the fact that some programming languages, such as Java, do not support multiple inheritance, the model defines two types of classes: ordinary and mixin. The mixin classes are represented in Java as interfaces rather than classes. The mixin classes are shown as circles rather than rectangles in the class diagram. It also shows the core classes in the Scope, Subject, and Artifact branches. It only shows the

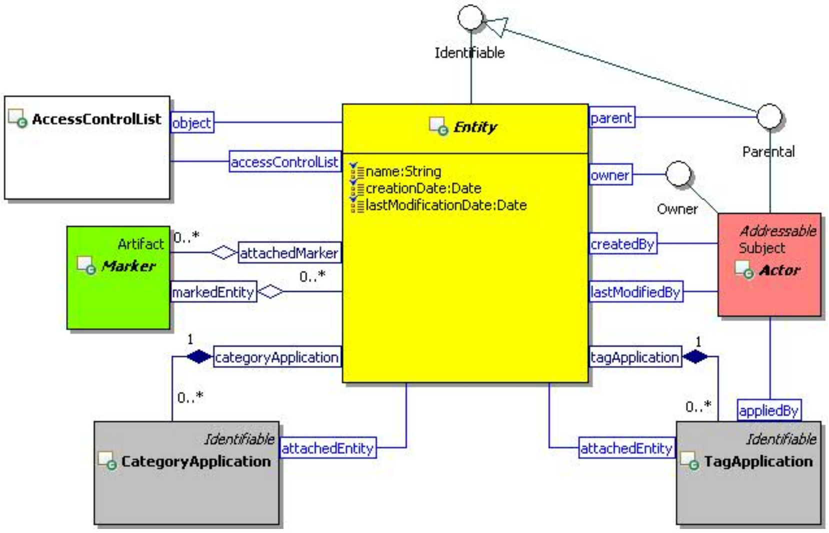


Fig. 2. ICOM Entity Classes

subclass relationships, not the attributes or the associations. Figure 2 depicts the Entity class in more detail, showing both its attributes and associations with other classes.

2.2 ICOM Extensions

Each ICOM extension module defines a model of a collaboration activity. The specification includes models for content creation, communication, coordination, discussion forum, and conference. Most of the extension modules in this section introduce specialized subclasses of Artifact and Folder of Artifact Branch. In addition to the extension modules described here, the ICOM framework allows additional extension modules. For example, applications can adopt a model for the CMIS Policy base type as a new extension module, which can be used to integrate with BPMN or BPEL processes outside the ICOM domain. An ICOM space can provide a durable context for continuity of conversations and activities related to a business process type or process instance. Some new extension modules may import the models from related standards.

ICOM defines containers that provide contexts and structures for specific areas of collaborative activities. For example, a Space is a container that contains HeterogeneousFolder, AddressBook, Calendar, TaskList, Forum, and Conference. These subcontainers are briefly described below in the corresponding extension modules. HeterogeneousFolder (defined in Core) is a general purpose container that can list any type of artifacts, and therefore, can serve as a library of documents and wiki pages to support content sharing and co-creation, an inbox or outbox for communication, or a trash folder to archive all types of artifacts deleted from a space.

The following modules are specified as extension modules of ICOM:

Content module defines classes for Content, MultiContent, and SimpleContent. Content represents a piece of data in a document or message. The module uses the Composite design pattern to form objects. The module is referenced by other extension modules, like the Document module and the Message module.

Document module defines Document, WikiPage, and a model for version control. A document can contain a composite content. Documents are typically contained by heterogeneous folders.

Message module defines Messages like emails or instant message, and related classes. A message can contain a composite content. Messages are typically contained by heterogeneous folders.

Presence module defines Presence, Activity, and ContactMethod. Presence represents a watchable state of an actor. Presence state is derived using an actor's subscriptions.

AddressBook module defines AddressBook and PersonContact. AddressBook is a specialized container to manage contact or personal information, such as addresses, phone numbers, birthdays, anniversaries, and other entries. A person contact can reference a person in an ICOM community as well as information about a person who may not be in any ICOM community.

Calendar module defines Calendar, Occurrence, and OccurrenceSeries. Calendar is a specialized container to support time management. Occurrence artifacts are used to resolve the free-busy times of participants for scheduling of meetings and booking of resources.

FreeBusy module defines the FreeBusy class. FreeBusy is a view derived from occurrences in a calendar or a set of calendars using an actor's privileges.

TaskList module defines TaskList and Task. TaskList is a specialized container to support task coordination. Tasks are used to coordinate the assignment and progress of work.

Forum module defines Forum, Topic, Announcement, and DiscussionMessage. Topics, announcements, and discussions are used for threaded conversations. Moderators of a forum can prune, merge, or fork the discussion threads. Forum is a specialized container to support Topic sub-containers and Announcement sub-containers for time-sensitive communication.

Conference module defines Conference and related classes. Conference is a specialized container that provides a durable context for real-time interactions. A conference can contain visual, audio, and chat transcripts of the conference sessions. It also contains the current status, conference settings, past sessions, active session, and activity logs.

3 Features of the Model

3.1 Persistence

Maintaining data persistently is a necessity for nearly all software applications. Since the predominant storage technology is the relational model but the predominant software languages are object-oriented, it is necessary to have an object-relational mapping. An increasing number of applications are using the Java Persistence API (JPA) for this task. Accordingly, one of the first mappings for ICOM was a mapping to JPA. This mapping consists of Java annotations added to the POJO classes. The java.net OpenICOM project [19] is incubating a JPA framework which manages ICOM POJO objects. OpenICOM framework emulates the JPA programming model but is not limited to object-relational mapping. It supports pluggable data access connectors that can be implemented using proprietary APIs for collaboration platforms, standard protocols such as LDAP, WebDAV, IMAP, SMTP, XMPP, CalDAV, etc., or NoSQL databases.

The ICOM specification defines a class called Entity which is the superclass of any class that supports a persistent identifier, a change token for optimistic locking, and an access control list. The object identifier and change token are annotated, respectively, by javax.persistence.Id and javax.persistence.Version, matching the ICOM concept of Entity with the JPA concept of Entity. ICOM Entity has another fundamental dimension for access control list, which together with JPA Id and Version, defines a unit of persistent information for concurrency and access control. The generation of object identifiers is implementation dependent; however, ICOM recommends that the object identifiers should be globally unique to support permanent references to the entities that may migrate amongst interoperable ICOM repositories. An object identifier is read-only (immutable) once it is assigned and should never be duplicated or re-used for more than one object. The UML diagram in Figure 2 depicts the Entity class, properties, and cardinality of the properties. Entity's properties include name, created by, creation date, last modified by, last modification date, owner, parent, attached markers, category applications, tag applications, and access control list.

Figure 3 shows some of JPA annotations for the Entity class. For the sake of brevity, the import statements were not shown.

3.2 Interoperability

One of the goals of ICOM is seamless interoperability between different collaborative work environments. This requires an interchange mechanism between such environments. Since these environments may use different programming languages, the interchange format must be language-independent. The most commonly-used interchange format today is XML, and ICOM has been expressed in terms of XML Schema. This allows one to exchange data via SOAP/REST web services. The XML Schema specification was derived from the POJO classes

```
package icom;

@Entity
@XmlType(name="Entity", namespace="http://docs.oasis-open.org/ns/icom/core/201008")
@XmlRootElement(namespace="http://docs.oasis-open.org/ns/icom/core/201008")
@XmlAccessorType(FIELD)
public abstract class Entity implements Identifiable {
 @EmbeddedId
 protected Id objectId;
 @Version
 protected ChangeToken changeToken;
 @ManyToOne
 Actor createdBy;
 @ManyToMany(targetEntity=Marker.class)
 @XmlElement(name="attachedMarker",
  namespace="http://docs.oasis-open.org/ns/icom/metadata/201008")
 Set<Marker> attachedMarkers;
 ...
}
```

Fig. 3. Some of the fields and their JPA annotations for the Entity class

```
<xs:complexType name="Entity" abstract="true">
 <xs:sequence>
  <xs:element name="objectId" type="icom_core:Id" minOccurs="0" />
  <xs:element name="changeToken" type="icom_core:ChangeToken" minOccurs="0" />
  <xs:element name="name" type="xs:string" minOccurs="0" />
  <xs:element name="creationDate" type="xs:dateTime" minOccurs="0" />
  <xs:element name="createdBy" type="icom_core:Actor" minOccurs="0" />
  <xs:element name="lastModificationDate" type="xs:dateTime" minOccurs="0" />
  <xs:element name="lastModifiedBy" type="icom_core:Actor" minOccurs="0" />
  <xs:element name="parent" type="xs:anyType" minOccurs="0" />
  <xs:element ref="icom_ac:owner" minOccurs="0" />
  <xs:element ref="icom_ac:accessControlList" minOccurs="0" />
  <xs:element ref="icom_meta:attachedMarker" minOccurs="0" maxOccurs="unbounded" />
  <xs:element ref="icom_meta:categoryApplication" minOccurs="0" maxOccurs="unbounded" />
  <xs:element ref="icom_meta:tagApplication" minOccurs="0" maxOccurs="unbounded" />
 </xs:sequence>
</xs:complexType>
```

Fig. 4. XML Schema for the Entity class

using the JAXB `schemagen` processor[9]. Figure 4 shows the XML Schema representation of the Entity class.

3.3 Semantic Representation

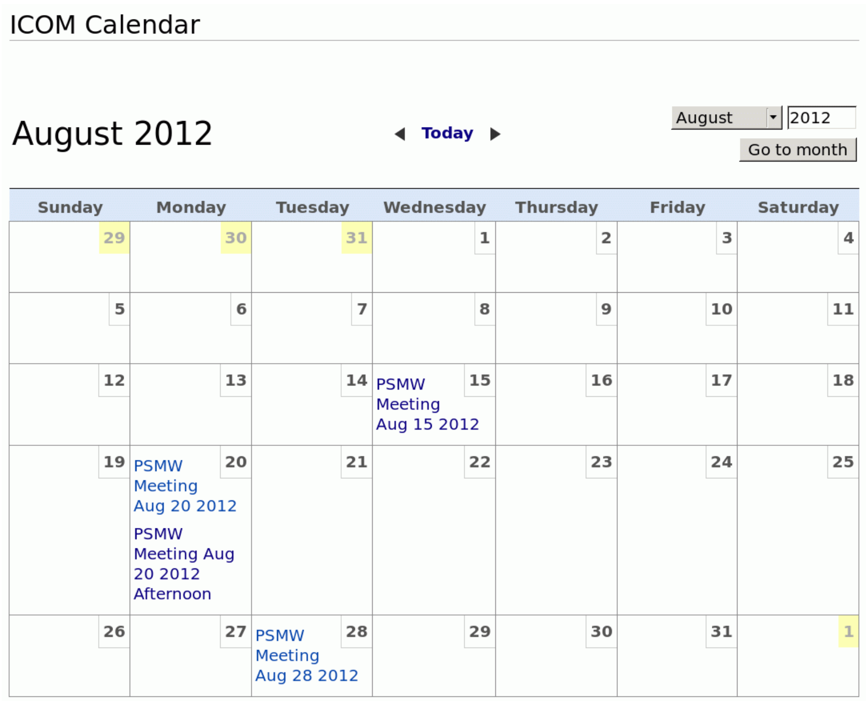
ICOM also has representations in RDF and OWL. The semantic representation is modular, reflecting the UML model, consisting of a core ontology, and extensions. The ICOM ontologies were defined through a direct translation from the authoritative specification.

Providing a semantic representation further accelerates the communication and interconnection of data between different collaboration tools. Other benefits include: access to the growing amount of Linked Data that is available, and inference. With inference, we can enrich the data, but also check for consistency when

[9] http://docs.oracle.com/javase/7/docs/technotes/tools/share/schemagen.html

using ICOM with other ontologies. ICOM data with a seamless programming model like JPA and a concomitant RDF representation will lower the barrier for applying inference engines. Figuratively speaking, a rich vocabulary of "nouns" in ICOM makes up for the strong "verbs" in service interfaces. A well-defined set of classes of ICOM makes the API amenable for rule-based applications and declarative inference. ICOM containers are active or reactive entities, for example conference and chat rooms are highly active while outbox, calendar, and task list are reactive. Their behavior can be augmented by applications.

4 A Use Case — Integration with Semantic Media Wiki



August 2012

Sunday	Monday	Tuesday	Wednesday	Thursday	Friday	Saturday
29	30	31	1	2	3	4
5	6	7	8	9	10	11
12	13	14	15 PSMW Meeting Aug 15 2012	16	17	18
19	20 PSMW Meeting Aug 20 2012 PSMW Meeting Aug 20 2012 Afternoon	21	22	23	24	25
26	27	28 PSMW Meeting Aug 28 2012	29	30	31	1

Fig. 5. Example of a calendar module that integrates SMW and ICOM

A *wiki* is a website which allows its users to collaborate to produce and edit content via a web browser. The resulting content is easily understood by users, but computers cannot understand or evaluate it. A *semantic wiki* allows a user to add semantic annotations that allow the wiki to function as a knowledge base which supports inference and semantic query capabilities. The most popular wiki software is MediaWiki[10], the underlying software for Wikipedia. The most popular semantic wiki software is Semantic MediaWiki (SMW)[11].

Unfortunately, just having semantic annotations does not mean that different individuals or communities will specify annotations that are compatible with other individuals or communities. Ontologies can provide standard terminologies for annotations that allow for semantic interoperability within an SMW site as well as between sites. The ICOM ontology (available in both RDF and OWL) is especially appropriate for SMW because both ICOM and SMW are concerned with supporting collaboration. We have developed a proof of concept for the use of ICOM in a semantic wiki by developing SMW modules for calendars, which make use of the powerful semantic search feature of SMW. Figure 5 illustrates one view of the calendar module. This module is publicly avalable at `http://psmw-test.cim3.net/w/index.php/ICOM_Calendar`.

Other capabilities enabled by the integration of ICOM with SMW include the ability to export data to different formats and to mash-up calendar data from multiple sources. Our SMW modules illustrate how the ICOM RDF and OWL ontologies can enhance interoperability of collaboration tools.

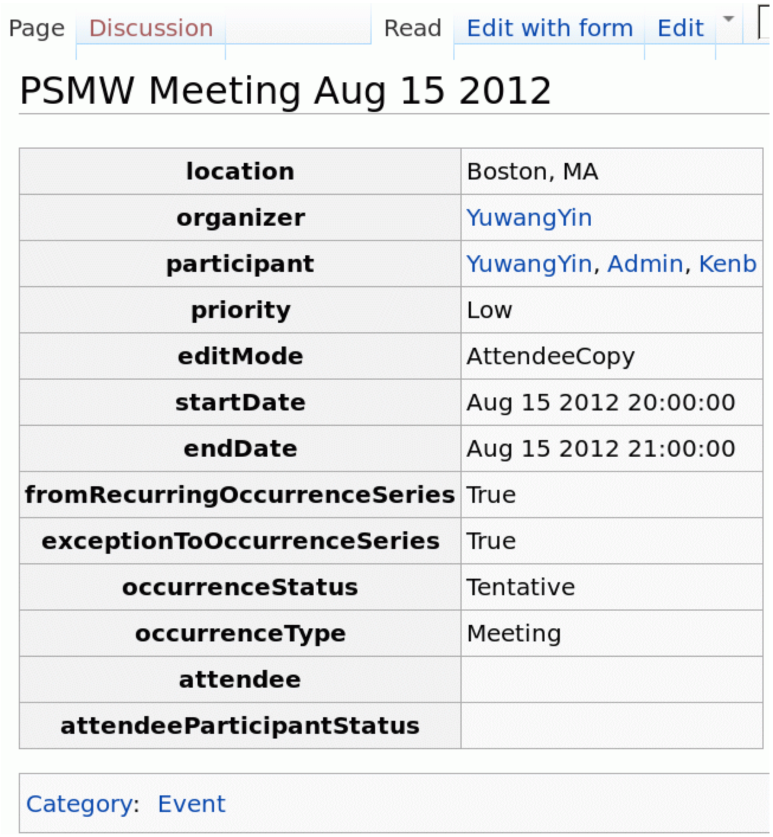
Our Calendar module is a customizable Template that is written in the calendar format of the Semantic Result Formats[20]. This Template uses ICOM properties. Instantiating the Template produces a reusable Form. If one clicks on the meeting on August 15 in Figure 5, one can see an example of such a Form, which is shown in Figure 6. This form shows the ICOM properties of the meeting along with their values. The values can be modified by clicking on the “Edit with form” tab. The meetings to be included in a particular Form are determined by a SPARQL[21] query. A user can either customize the Template or reuse an existing Form to create their own Calendar.

5 Conclusion and Future Work

We have described the ICOM framework for Interoperable Collaboration Services. Like most standards, the ICOM is specified in an authoritative document using a specification language (CMIS) that is designed to be easily readable by humans. However, ICOM is intended to be mainly used as an interoperability framework within a collaborative work environment for such tasks as storing data, sending data to other collaboration services as well as interacting with human collaborators. As a result, the ICOM framework has been translated to a number of other languages to facilitate these many purposes. Inasmuch as many other kinds of applications have these same requirements, it would be useful for their standards to be translated to other languages as well. Accordingly, we plan to use our translation software with other standards.

[10] http://www.mediawiki.org/

[11] http://semantic-mediawiki.org/



PSMW Meeting Aug 15 2012

location	Boston, MA
organizer	YuwangYin
participant	YuwangYin, Admin, Kenb
priority	Low
editMode	AttendeeCopy
startDate	Aug 15 2012 20:00:00
endDate	Aug 15 2012 21:00:00
fromRecurringOccurrenceSeries	True
exceptionToOccurrenceSeries	True
occurrenceStatus	Tentative
occurrenceType	Meeting
attendee	
attendeeParticipantStatus	

Category: Event

Fig. 6. An example of a customized Form used for a meeting

Acknowledgements

The authors wish to extend their thanks to colleagues who have helped with different domain expertise. Especially, we thank Rafiul Ahad and Stefan Decker for supporting this work.

This is a post-peer-review, pre-copyedit version of an article published in: A. Haller et al. (Eds.): WISE 2011 and 2012 Combined Workshops, LNCS 7652, pp. 147–158. Springer, Heidelberg (2013).

References

1. Zeilenga, K.D.: Lightweight Directory Access Protocol (LDAP): Directory Information Models. IETF RFC 4512 (Proposed Standard) (June 2006)
2. Brown, A., Gur-Esh, E., McVeigh, R., Müller, F.: Content Management Interoperability Services (CMIS), Version 1.0. OASIS Standard (May 2010)

3. Nuescheler, D.: Content Repository for Java^{TM} Technology API Version 2.0. http://jcp.org/en/jsr/detail?id=283 (August 2009)
4. Dusseault, L.: HTTP Extensions for Web Distributed Authoring and Versioning (WebDAV). IETF RFC 4918 (Proposed Standard) (June 2007)
5. Crispin, M.: Internet Message Access Protocol Version 4rev1. IETF RFC 2060 (Proposed Standard) (December 1996)
6. Klensin, J.: Simple mail transfer protocol. IETF RFC 5321 (Draft Standard) (October 2008)
7. Saint-Andre, P.: Extensible Messaging and Presence Protocol (XMPP): Core. IETF RFC 3920 (Proposed Standard) (October 2004) Obsoleted by RFC 6120, updated by RFC 6122.
8. Saint-Andre, P.: Extensible Messaging and Presence Protocol (XMPP): Instant Messaging and Presence. IETF RFC 3921 (Proposed Standard) (October 2004) Obsoleted by RFC 6121.
9. Dawson, F., Howes, T.: vCard MIME Directory Profile. IETF RFC 2426 (Proposed Standard) (September 1998)
10. Desruisseaux, B.: Internet Calendaring and Scheduling Core Object Specification (iCalendar). IETF RFC 5545 (Proposed Standard) (September 2009)
11. Daboo, C., Desruisseaux, B., Dusseault, L.: Calendaring Extensions to WebDAV (CalDAV). IETF RFC 4791 (Proposed Standard) (March 2007)
12. : Business Process Model and Notation (BPMN) Version 2.0. http://www.omg.org/spec/BPMN/2.0 (January 2011)
13. Jordan, D., Evdemon, J.: Web Services Business Process Execution Language (WSBPEL), Version 2.0. OASIS Standard (April 2007)
14. Ings, D.: WS-BPEL Extension for People (BPEL4People) Specification, Version 1.1. OASIS Committee Specification (August 2010)
15. Ings, D.: Web Services — Human Task (WS-HumanTask) Specification, Version 1.1. OASIS Committee Specification (August 2010)
16. Brickley, D., Miller, L.: FOAF Vocabulary Specification 0.98. http://xmlns.com/foaf/spec/ (August 2010)
17. Berrueta, D., Brickley, D., Decker, S., Fernndez, S., Grn, C., Harth, A., Heath, T., Idehen, K., Kjernsmo, K., Miles, A., Passant, A., Polleres, A., Polo, L., Sintek, M.: SIOC Core Ontology Specification. W3C Member Submission, W3C (June 2007)
18. Chan, E.S., Durusau, P.: Integrated Collaboration Object Model (ICOM) for Interoperable Collaboration Services, Version 1.0. OASIS Committee Specification (August 2012)
19. Chan, E.S.: OpenICOM: A JPA Framework for Integrated Collaboration Environments, Part 1. http://today.java.net/article/2011/03/21/openicom-jpa-framework-integrated-collaboration-environments-part-1 (March 2011)
20. Hong Kong, J., Koren, Y., De Dauw, J.: Semantic Result Formats. http://semantic-mediawiki.org/wiki/Semantic_Result_Formats (2012)
21. Prud'hommeaux, E., Seaborne, A.: SPARQL Query Language for RDF. W3C Recommentation, W3C (January 2008)