Okkam*4P*: A Protégé Plugin for Supporting the Re-use of Globally Unique Identifiers for Individuals in OWL/RDF Knowledge Bases[*]

Paolo Bouquet[1], Heiko Stoermer[1], and Xin Liu[2]

[1] University of Trento,
Dept. of Information and Communication Tech.,
Trento, Italy
{bouquet, stoermer}@dit.unitn.it
[2] JiLin University
Dept. of Computer Science and Technology
Chang Chun, China
xinliujlu@gmail.com

Abstract. In Protégé, any newly created RDF/OWL knowledge base refers to local instances through a *local* URI, which is obtained through the concatenation of the ontology URI, the hash sign # and a local identifier. However, this practice makes data-level integration quite hard, and definitely prevents the straightforward application of RDF graph merging for independently developed knowledge bases, even if they share the same OWL ontology. In this paper, we present a Protégé plugin which supports the systematic reuse of global identifiers for instances in RDF/OWL knowledge base. The plugin is an extension of the Protégé "Individuals" tab. The main difference is that, when an instance is created, the user has a chance of looking for an existing URI for the corresponding individual in a publicly available service called Okkam. The match between the newly created instance and the globally registered individuals is based on a comparison of features of the new and a a simple profile stored in Okkam for all individuals. The plugin is available and tested for Protégé 3.3.1 and 3.4 beta.

1 Introduction

One of the key ideas of the Semantic Web is that the use of a unique identifier (URI) for referring to the same resource will be the basis for enabling the integration of data across autonomous applications and independently created semantic repositories. However, nothing in the infrastructure of the Semantic Web supports content creators to reuse already existing URIs for referring to a

[*] This work was partially funded by the European Commission under the 6th Framework Programme IST Integrated Project VIKEF - Virtual Information and Knowledge Environment Framework (Contract no. 507173, Priority 2.3.1.7 Semantic-based Knowledge Systems; more information at http://www.vikef.net). The authors would like to thank Daniele Zanoni for his work on the first prototype of Okkam*4P*.

resource which has already been referred to in other applications/repositories. This is true for any type of resource (including abstract resources, like classes and properties), but is especially bad for instances (individuals), as the *ex-post* discovery of identities between instances across knowledge bases is in general more difficult (and less investigated) than discovering mappings between ontology elements. The lack of systematic support for the reuse of URIs leads to a flooding of identifiers, which makes data integration on the Semantic Web very hard and error prone.

The issue of identity and identification in the (Semantic) Web has been discussed and analyzed from different perspectives in the past, in fact two scientific workshops have been dedicated to the topic[3], the proceedings of which are a great source of insight into the diverse points of view on the issue[4]. Additionally, works such as Kent's [9, 10] from a historical perspective, Gangemi and Presutti's [6, 7] as well as the efforts and discussions within the W3C [2, 1, 8], make evident that there is plenty of ambiguity about the use and semantics of a URI. There are several options of what a URI can identify, opinions about whether a URI should be de-referenceable or not (and how), how syntactically URIs should be constructed that refer to non-electronic objects, and – last but not least – the intuition that the uniqueness property of URIs which their name suggests is desirable, but in no way guaranteed.

This last point is the motivation of our work. Based on the availability of an open public service for supporting the global reuse of unique identifiers for individual instances called OKKAM [3], which is described in Sect. 2, we are developing tool support for content creation. The global service is based on an open public repository which stores previously created identifiers for individuals, together with a simple profile. The idea is that this service can be used to look up for pre-existing identifiers of any newly created instance in a knowledge base; this process is based on an entity matching algorithm, which uses any available information about the new entity to match it with the profiles of individuals stored in the repository and thus to find candidate URIs for reuse.

The application we are going to present makes use of this service in the area of ontology editing. It aims at demonstrating the advantages of such an approach as a way to converge on common URIs for newly created semantic content. Indeed, a common practice in ontology editing is the creation of new (local) URIs for any newly created instance. Here we present a Protégé plugin, named OKKAM*4P*, which supports the good practice of looking up for pre-existing URIs when editing a new RDF/OWL knowledge base. The plugin is an extension of the "individual" tab. The main difference is that, when an instance is created, the user has a chance of looking for a pre-existing URI for the corresponding individual in a publicly available service called OKKAM. The match between the newly created instance and the stored individuals is based on an algorithm which compares the features of the new instance in the local knowedge base

[3] IRW in 2006 (`http://www.ibiblio.org/hhalpin/irw2006/`) and I^3 in 2007 (`http://okkam.dit.unitn.it/i3/`)

[4] see [5] for I^3 and the workshop website for IRW

with the profiles stored in OKKAM. The plugin is available and tested for the latest official release of Protégé, version 3.3.1, and the beta version 3.4; the experimental OKKAM service is accessible at `http://www.okkam.org`.

2 The Okkam*PUBLIC* Infrastructure

The work described in this paper relies on the existance of the OKKAM infrastructure, the initial idea of which was described in more detail in [4, 3]. As illustrated in Figure 1, at the heart of this infrastructure there is the central repository for entity identifiers, called OKKAM*PUBLIC*[5]. This repository can be imagined like a very large catalog, where semi-structured descriptions of entities are stored and associated to globally unique identifiers for these entities. It furthermore provides the functionality to add entities and their descriptions to the repository that have not existed there so far, and to retrieve their OKKAM identifiers for use in information systems.

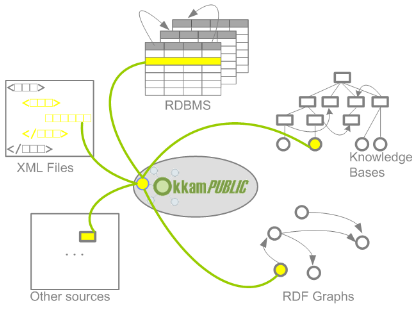


Fig. 1. Overview of the global OKKAM vision.

Figure 2 illustrates the standard use-case for the *okkamization*[6] of content, namely to query OKKAM*PUBLIC* for the existance of the entity at hand. This would usually be achieved through functionality provided by a client application – in this case Protégé – which accesses the OKKAM*PUBLIC* API, and presents (if available) a list of top candidates which match the description for the entity provided within the client application. If the entity is among these candidates, the client agent (human or software) uses the associated OKKAM identifier in the respective information object(s) *instead* of a local identifier. If the entity cannot be found, the client application can create a new entry for this entity in OKKAM and thus cause an identifier for the entity to be issued and used as described before.

[5] This service is currently under development at the University of Trento, and will be opened for public access in the near future.

[6] We call *okkamization* the process of assigning an OKKAM identifier to an entity that is being annotated in any kind of content, such as an OWL/RDF ontology, an XML file, or a database, to make the entity globally identifiable.

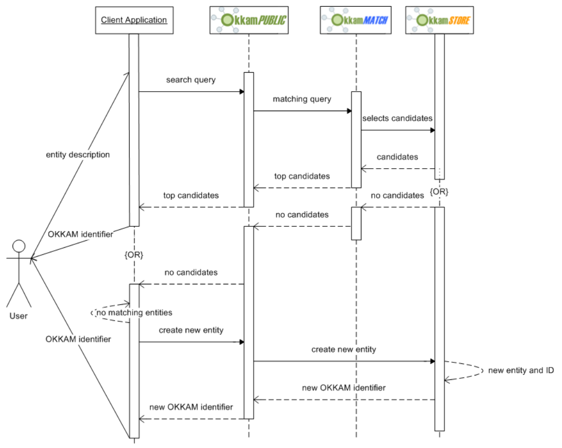


Fig. 2. Sequence diagram of the OKKAM standard use case.

The large-scale, global service OKKAM*PUBLIC* provides for the entity repository and a service infrastructure so that tools and applications can make use of this new technology. The current version of OKKAM*PUBLIC* is a prototypical implementation of parts of a larger multi-tier architecture, namely a non-distributed version of the storage component OKKAM*STORE* which in a later phase will move to a distributed layout, a preliminary version of the matching component OKKAM*MATCH* which performs the search for entities, and a subset set of the developer API and toolkit OKKAM*DEV* which is available[7].

The mechanisms inside OKKAM which perform the matching between entity descriptions provided by the user or agent and the existing descriptions stored in the repository, display some specifics which should be mentioned at this point. One of the main characteristics of OKKAM is that the description of an entity, which is necessarily used to distuingish this entity from all others in the repository, does *not* follow a fixed schema, i.e. OKKAM is specifically not something like a knowledge base of entities; consequently, OKKAM is not providing an ontological formalization of which attributes an entity has. The way to describe entities is extremely flexible and semi-structured, realised by way of key/value pairs which can contain arbitrary strings. The reasons for this decisions have been laid out in [4, 3], and basically go back to the point that there is an infinite variety of ways of how to model domains, for which reasons we decided to stay completely domain independent. As a consequence, the matching algorithms in OKKAM*MATCH* can take as input *any* kind of description of an entity, e.g. the set of properties and values inferred from an ontology, and match it against existing data. This is how we achieve OKKAM support without any dependence on, or knowledge of, an underlying schema.

[7] `http://www.okkam.org`

3 Okkam*4P* – Making Protégé an Okkam-empowered Tool

3.1 User Perspective

In our vision of a functioning OKKAM infrastructure there is the notion of the so-called "OKKAM-empowered tools", which are standard end-user applications (e.g. word processors, HTML/XML/OWL editors, web-based authoring environments – like blogs, forums, multimedia publishing and tagging applications, etc.) extended with functionalities which facilitate the creation of okkamized content through the use of the OKKAM*PUBLIC* infrastructure. Protégé falls into this category. It is probably the most widely used editor for the creation of RDF/OWL knowledge bases (KBs), and provides vast extensibility through a plugin architecture, which makes it highly suitable for empowering it with OKKAM functionality.

The plugin presented in this paper essentially assigns a global unique identifier called (the "OKKAM ID") to a newly created individual, rather than relying on manual input of the user or the standard automatic mechanism of Protégé. To this end, it implements the use-case illustrated in Fig. 2: based on the data about an individual that are already provided in the KB developed by the user, it queries OKKAM*PUBLIC* to see whether an identifier already exists which can be assigned to the new created individual, otherwise a new identifier would be created.

To use this plugin, the user selects an individual and right-clicks on it. A context menu will pop up, in which the item "Get Okkam ID" is the entry-point to the functions of the plugin, as illustrated in Fig. 3.

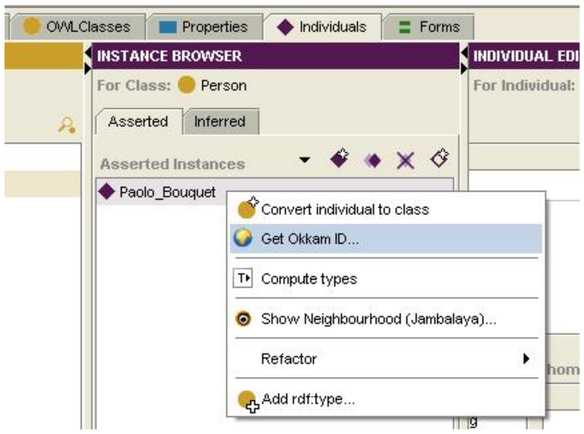
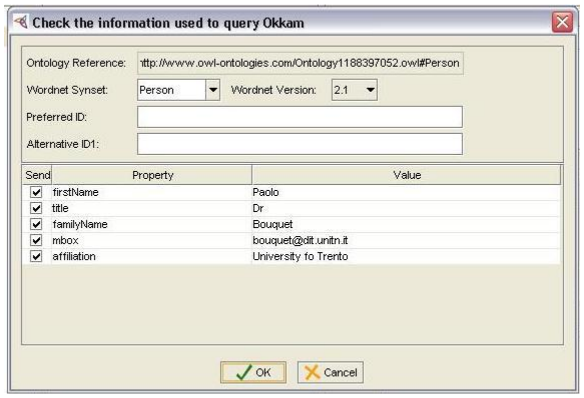


Fig. 3. Assigning a global identifier to an individual.

Once clicking on this menu, the plugin starts to collect the properties of this individual as specified in the KB, and presents a new dialog (see Fig. 4) displaying the information that is available for querying OKKAM*PUBLIC* in order to see whether an identifier for this entity already exists.



Ontology Reference: ıttp://www.owl-ontologies.com/Ontology1188397052.owl#Person
Wordnet Synset: Person
Wordnet Version: 2.1
Preferred ID:
Alternative ID1:

Send	Property	Value
✓	firstName	Paolo
✓	title	Dr
✓	familyName	Bouquet
✓	mbox	bouquet@dit.unitn.it
✓	affiliation	University fo Trento

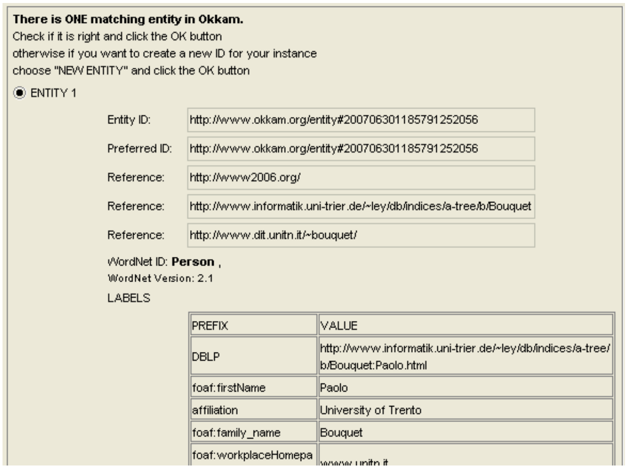
Fig. 4. The information of the chosen individual.

The properties that are gathered by the plugin to construct a query are the following:

- Ontology Reference: it is the reference of the ontology which the chosen individual belongs to. It is loaded automatically by this plugin, and it is read-only for users. If the ontology is publicly available, it can potentially be of use for the server-side matching mechanisms to improve search results for the individual.
- Wordnet Synset and Wordnet Version: provides a hint about a top-level class which the chosen individual belongs to. This has to be set by the user.
- Preferred ID and Alternative ID1: if the user wishes to use another identifier in other systems to identify the chosen individual, a user can input this identifier here. These two items are optional.
- Individual Properties: the plugin loads each property of the chosen individual automatically. The user can also deselect some properties which are thought to be unnecessary to find the OKKAM ID of the individual at hand.

After submitting this form, the plugin launches a thread to query OKKAM*PUBLIC* for matching entities by calling its web service. After searching, a list of entities that match the description for the new created individual will be visualized to the user, as illustrated in Fig. 5

The user now has the option to select one list entry as "the same" as the newly created individual and re-use the global identifier in the local KB (therefore the ID of the newly created individual will be replaced by the OKKAM ID in the



There is ONE matching entity in Okkam.
Check if it is right and click the OK button
otherwise if you want to create a new ID for your instance
choose "NEW ENTITY" and click the OK button

● ENTITY 1

Entity ID:	http://www.okkam.org/entity#200706301185791252056
Preferred ID:	http://www.okkam.org/entity#200706301185791252056
Reference:	http://www2006.org/
Reference:	http://www.informatik.uni-trier.de/~ley/db/indices/a-tree/b/Bouquet
Reference:	http://www.dit.unitn.it/~bouquet/

WordNet ID: **Person** ,
WordNet Version: 2.1
LABELS

PREFIX	VALUE
DBLP	http://www.informatik.uni-trier.de/~ley/db/indices/a-tree/b/Bouquet:Paolo.html
foaf:firstName	Paolo
affiliation	University of Trento
foaf:family_name	Bouquet
foaf:workplaceHomepa	www.unitn.it

Fig. 5. Query result of with matching entities that already have an identifier in Okkam.

KB); otherwise the user can choose to create the individual as a new entity in Okkam*PUBLIC*, in which case the information selected in Fig. 4 will be inserted into Okkam repository, the new Okkam ID will be retrieved and assigned to the local individual.

3.2 Developer Perspective

The hierarchy of primary classes provided by and used in this plugin is illustrated in Fig. 6 in the appendix. In the following we describe the function of each class displayed in Fig. 6.

The class *OkkamPlugIn* is the most principal class. To extend the "Individuals" tab in protege, it needs to inherit the class
edu.stanford.smi.protegex.owl.ui.actions.ResourceAction. This effects that the menu item "Get Okkam ID..." will appear in context-menu when the user right-clicks on a individual.

The class *okkamPanel* and *TopPanel* are used to compose the information window (see Fig. 4); the class *ResultPanel* is used to show the query result window(see Fig. 5). All of them inherit the class *javax.swing.JPanel* to present a window to users.

In this plugin, we make use of web services to interact with Okkam*PUBLIC*. The tasks of searching for matching entities and publishing a newly created entity are fulfilled by calling the webservice "EntitySearch" and "EntityPublicationWithURI" respectively. These webservices are reachable from the URL `http://okkam.dit.unitn.it:8081/OkkamCoreWebServices/services`. As complex queries can have a considerable runtime, in the initial version of Okkam*4P*, users would see nothing but a gray window until the result returned from the webservice. In the current version, we moved the plugin to a multi-threaded architecture. Three classes which inherit class "java.lang.Thread" are new to this version.

The class *InquireThread* is used to call the webservice "EntitySearch", it is launched when the user submits the information to search for matching entities. The class *PublishThread* is used to call the webservice "EntityPublicationWithURI", it is launched when the user decides to publish a new entity to OKKAM*PUBLIC*. The class *DialogThread* is used to show a dialog during the process of searching or publishing, this dialog is meant as a user-friendly interface to inform the users that the process is running.

4 Benefits of the Approach

The vision of the OKKAM approach is the creation of what we call the *Web of Entities* (WoE): a global information space in which entities (as opposed to documents) are the main objects of discourse and thus the pivot for information access.

The pre-requisite for this WoE to function is the existence of suitable *okkamized* content, i.e. content in which identified entities (such as persons, events, locations, ...) are denoted by their globally unique OKKAM identifier, instead of a local identifier, as described in the introduction.

To achieve a substantial diffusion of okkamized content, a set of user-friendly OKKAM-empowered tools is necessary, because – as the rather slow adoption of Semantic Web technologies has shown – the mass of content creators (i.e. the users of the WWW) seem not to be extremely motivated to follow developments beyond the coding of HTML documents.

With OKKAM*4P* we are making the first and very important step towards the creation of such a suite of tools. We address the community that is "closest" to the issues addressed by the approach, and provide them with the means of creating okkamized RDF/OWL KBs. The aim is to prove that – with the systematic a-priori use of global identifiers for entities – the vision of RDF documents as a single, global, *decentralized* and *meaningful* knowledge base can in fact become reality, without having to deal with many of the difficulties of information integration, such as the ex-post alignment of entities.

5 Future Work and Conclusion

In this paper we have presented our ongoing work on OKKAM*4P*, a plugin for the creation of *okkamized* RDF/OWL knowledge bases in Protégé, and given a sketch of the underlying, globally available infrastructure OKKAM*PUBLIC*.

As regards the plugin, several improvements are scheduled in the near future. One is the general "elevation" of the tool to a more production-quality standard, including the usual aspects such as extended documentation, code improvements, etc. Secondly, as the plugin is currently implemented as an extension to the OWL part of Protégé, KBs developed in plain RDF(S) cannot benefit from its functionality – a circumstance which we are currently investigating. Finally, additional features such as offline and batch operation, as well as automatic retrieval and assignment of Okkam identifiers to existing KBs, are already in the design phase.

Okkam*PUBLIC* itself will experience a great boost in the course of the European FP7 Integrated Project Okkam, which has the aim and the means to implement the infrastructure briefly illustrated in Sect. 2 at a very large scale.

More information will be made available at `http://www.okkam.org`, the plugin itself is available from `http://www.okkam.org/projects/okkam4p/`.

References

[1] T. Berners-Lee, R. Fielding, and L. Masinter. *RFC 3986: Uniform Resource Identifier (URI): Generic Syntax*. IETF (Internet Engineering Task Force), 2005. http://www.ietf.org/rfc/rfc3986.txt.

[2] Tim Berners-Lee. Design Issues – Linked Data. Published online, May 2007. `http://www.w3.org/DesignIssues/LinkedData.html`.

[3] Paolo Bouquet, Heiko Stoermer, and Daniel Giacomuzzi. OKKAM: Enabling a Web of Entities. In *i3: Identity, Identifiers, Identification. Proceedings of the WWW2007 Workshop on Entity-Centric Approaches to Information and Knowledge Management on the Web, Banff, Canada, May 8, 2007.*, CEUR Workshop Proceedings, ISSN 1613-0073, May 2007. online http://CEUR-WS.org/Vol-249/submission_150.pdf.

[4] Paolo Bouquet, Heiko Stoermer, Michele Mancioppi, and Daniel Giacomuzzi. OkkaM: Towards a Solution to the "Identity Crisis" on the Semantic Web. In *Proceedings of SWAP 2006, the 3rd Italian Semantic Web Workshop, Pisa, Italy, December 18-20, 2006. CEUR Workshop Proceedings, ISSN 1613-0073, online http://ceur-ws.org/Vol-201/33.pdf*, December 2006.

[5] Paolo Bouquet, Heiko Stoermer, Giovanni Tummarello, and Harry Halpin, editors. *i3: Identity, Identifiers, Identification. Proceedings of the WWW2007 Workshop on Entity-Centric Approaches to Information and Knowledge Management on the Web, Banff, Canada, May 8, 2007.*, volume 249 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2007. online http://CEUR-WS.org/Vol-249/.

[6] Aldo Gangemi and Valentina Presutti. Towards an OWL Ontology for Identity on the Web. In *Semantic Web Applications and Perspectives (SWAP2006)*, 2006.

[7] Aldo Gangemi and Valentina Presutti. A grounded ontology for identity and reference of web resources. In *i3: Identity, Identifiers, Identification. Proceedings of the WWW2007 Workshop on Entity-Centric Approaches to Information and Knowledge Management on the Web, Banff, Canada, May 8, 2007.*, 2007.

[8] Ian Jacobs and Norman Walsh. Architecture of the world wide web, volume one. Published online, December 2004. `http://www.w3.org/TR/webarch/`.

[9] William Kent. The Entity Join. In *Fifth Intl. Conf. on Very Large Data Bases, Rio de Janeiro, Brazil*, pages 232–238. Morgan Kaufman Publishers, 1979.

[10] William Kent. A Rigorous Model of Object Reference, Identity, and Existence. *Journal of Object-Oriented Programming*, 4(3):28–38, June 1991.

Appendix: Okkam*4P* Class Diagram

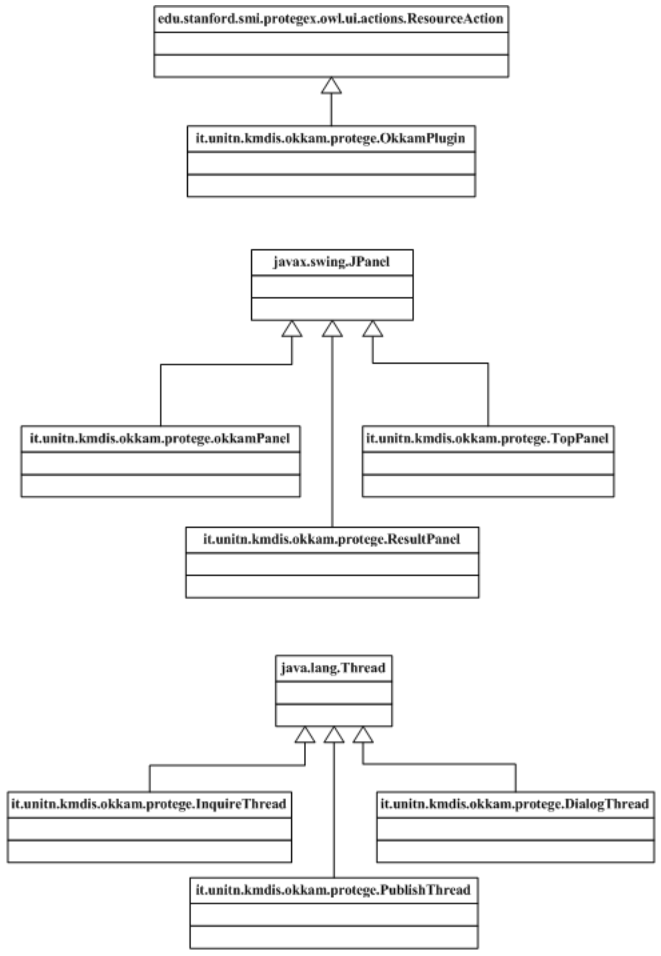

edu.stanford.smi.protegex.owl.ui.actions.ResourceAction
it.unitn.kmdis.okkam.protege.OkkamPlugin
javax.swing.JPanel
it.unitn.kmdis.okkam.protege.okkamPanel
it.unitn.kmdis.okkam.protege.TopPanel
it.unitn.kmdis.okkam.protege.ResultPanel
java.lang.Thread
it.unitn.kmdis.okkam.protege.InquireThread
it.unitn.kmdis.okkam.protege.DialogThread
it.unitn.kmdis.okkam.protege.PublishThread

Fig. 6. UML class diagram showing the primary classes of OKKAM*4P*.